

# mcBrief: un descriptor local de features para imágenes color

Tesina de grado

Autor: Daniel Moreno

Director: Mario E. Munich

Julio 2011



*Lic. en Cs. de la Computación  
Facultad de Ciencias Exactas, Ingeniería y Agrimensura  
Universidad Nacional de Rosario*

# Resumen

En *Computer Vision*, una tarea habitual es la de reconocimiento de imágenes. Este tipo de tarea puede resolverse obteniendo muy buenos resultados empleando descriptores de features locales, como SIFT, SURF o BRIEF; pero estos descriptores no están diseñados para trabajar con imágenes color. Sin embargo, usualmente las tareas de reconocimiento se realizan con este tipo imágenes, las que deben previamente convertirse al formato de escala de grises, perdiendo parte de la información original y potencialmente útil, para luego proceder a la extracción de descriptores. En este trabajo se propone un nuevo descriptor de features locales, denominado `mcBrief`, que puede extraerse en forma directa desde imágenes a color, aprovechando mejor la información que éstas contienen y habitualmente se desecha. Las pruebas hechas muestran que el nuevo descriptor mejora el resultado de las tareas de reconocimiento y es igual de eficiente, en términos de espacio y velocidad, que sus pares en el estado del arte.

# Índice general

<b>1. Introducción</b>	<b>1</b>
<b>2. Conocimientos generales</b>	<b>3</b>
2.1. Representación de imágenes . . . . .	3
2.2. Reconocimiento de imágenes . . . . .	4
2.3. Propiedades de un descriptor ideal . . . . .	4
2.4. Modelo diagonal . . . . .	5
2.4.1. Cambio de intensidad . . . . .	6
2.4.2. Desplazamiento de intensidad . . . . .	6
2.4.3. Cambio y desplazamiento de intensidad . . . . .	6
2.4.4. Cambio de color . . . . .	7
2.4.5. Cambio y desplazamiento de color . . . . .	7
2.5. Transformaciones geométricas en 2D . . . . .	7
2.5.1. Traslación . . . . .	8
2.5.2. Rotación + traslación . . . . .	8
2.5.3. Transformación afín . . . . .	8
2.5.4. Transformación proyectiva . . . . .	8
2.5.5. Jerarquía de transformaciones 2D . . . . .	9
2.6. Imagen integral . . . . .	9
2.7. Evaluación de algoritmos de matching . . . . .	10
<b>3. Trabajos relacionados</b>	<b>12</b>
3.1. SIFT: Scale-Invariant Features . . . . .	12
3.2. SURF: Speeded Up Robust Features . . . . .	12
3.3. BRIEF: Binary Robust Independent Elementary Features . . . . .	13
3.4. Descriptores Color . . . . .	13
<b>4. Descriptor mcBrief</b>	<b>15</b>
4.1. Descripción . . . . .	15
4.1.1. Definición formal . . . . .	15
4.2. Complejidad . . . . .	16
4.3. Invariantes . . . . .	17
4.3.1. Ruido en la imagen y deformaciones afines . . . . .	17
4.3.2. Cambios en la iluminación de la escena . . . . .	17
4.3.3. Cambios de escala y rotación . . . . .	19
<b>5. Resultados experimentales</b>	<b>21</b>
5.1. Configuración de prueba . . . . .	21
5.2. Evaluación del descriptor . . . . .	21

5.2.1.	Descripción del dataset . . . . .	21
5.2.2.	Distribución de distancias . . . . .	22
5.2.3.	Precision vs. recall . . . . .	25
5.2.4.	Análisis ROC . . . . .	25
5.3.	Reconocimiento de imágenes . . . . .	27
5.3.1.	Dataset Caltech-Covers . . . . .	28
5.3.2.	Dataset Pasadena-Houses . . . . .	29
5.3.3.	Algoritmo de entrenamiento . . . . .	29
5.3.4.	Estrategia de matching . . . . .	30
5.3.5.	Resultados . . . . .	30
<b>6.</b>	<b>Conclusión y trabajos futuros</b>	<b>34</b>
<b>A.</b>	<b>Definición de tests binarios</b>	<b>36</b>

# Índice de figuras

2.1. Muestra de imágenes con diferentes alteraciones . . . . .	5
2.2. Transformaciones planares básicas . . . . .	7
2.3. Imagen integral . . . . .	10
2.4. Cálculo de sumas mediante una imagen integral . . . . .	10
2.5. Ejemplo de curva ROC . . . . .	11
4.1. Ejemplo de disposición espacial de los tests binarios sobre un patch de imagen. . . . .	16
5.1. Imágenes de referencia en el dataset de evaluación . . . . .	22
5.2. Muestra de la creciente dificultad de reconocimiento en los grupos Wall y Graf . . . . .	23
5.3. Histograma normalizado de distancias para el dataset Wall . . . . .	24
5.4. Gráficos 1-precision vs. recall . . . . .	26
5.5. Curva ROC consolidada . . . . .	27
5.6. Ejemplo del dataset Caltech-Covers . . . . .	28
5.7. Ejemplo del dataset Pasadena-Houses . . . . .	29

# Índice de tablas

2.1. Jerarquía de transformaciones 2D . . . . .	9
5.1. Valores de AUC y ERR . . . . .	27
5.2. Resultados para el dataset Caltech-Covers . . . . .	32
5.3. Resultados para el dataset Pasadena-Houses . . . . .	32
5.4. Tiempo promedio de extracción y comparación de descriptores . . . . .	32
A.1. Tests binarios mcBrief-32 (#1-#29) . . . . .	36
A.2. Tests binarios mcBrief-32 (#30-#75) . . . . .	37
A.3. Tests binarios mcBrief-32 (#76-#120) . . . . .	38
A.4. Tests binarios mcBrief-32 (#121-#166) . . . . .	39
A.5. Tests binarios mcBrief-32 (#167-#212) . . . . .	40
A.6. Tests binarios mcBrief-32 (#213-#256) . . . . .	41

# Índice de algoritmos

1.	Entrenamiento . . . . .	30
2.	Matching . . . . .	31

# Capítulo 1

## Introducción

La tarea de encontrar correspondencias entre dos imágenes de una misma escena es un problema común en el área de *Computer Vision*. Muchas aplicaciones dependen para su éxito de que este paso se resuelva correcta y eficientemente. Ejemplo de estas aplicaciones son el reconocimiento de objetos, la calibración de cámaras, la reconstrucción de escenas 3D y la localización visual (vSLAM). Además, no son pocas las veces en que las mismas se deben resolver bajo restricciones de tiempo real o se deben ejecutar en dispositivos con capacidades de cómputo limitadas, por lo cual, algoritmos más simples y más eficientes representan una ventaja importante.

La forma *estándar* de resolver este tipo de problemas consiste en caracterizar cada imagen mediante un conjunto de *descriptores locales* extraídos de ciertos puntos de ella y, en lugar de comparar las imágenes entre sí, se comparan los conjuntos de descriptores. Así, el problema de encontrar correspondencias entre imágenes, pasa a ser el problema de encontrar correspondencias entre descriptores, y el éxito en esta tarea depende en gran medida de las características del descriptor usado. El desafío es, entonces, la definición de descriptores con propiedades tales que permitan identificar correctamente correspondencias entre sectores de imágenes relacionadas, e identificar correctamente los casos en que un par de descriptores corresponden a imágenes no relacionadas.

Las imágenes con las que se trabaja corresponden, habitualmente, a fotografías tomadas cada una en un instante de tiempo diferente, lo que implica condiciones cambiantes. Condiciones que pueden ser propias del entorno, como sería la iluminación de la escena, o cambios producidos en la escena misma por el paso del tiempo; y condiciones que dependen del procedimiento de captura de la imagen, como lo es el tipo y configuración de la cámara empleada, y la posición de la misma, incluyendo su posición física y su orientación. A pesar de las marcadas diferencias que se observarán en estas imágenes, igualmente se pretende determinar correctamente la correspondencia entre ellas, responsabilidad que recae sobre el tipo de descriptor elegido. Debido a esto es que, en el caso ideal, un descriptor útil para esta tarea debería ser totalmente invariante a este tipo de diferencias presentes en cada imagen.

Si se leyó cuidadosamente el párrafo anterior, se habrá comprendido que las exigencias puestas sobre los descriptores locales son muchas y complejas, sin embargo, encontramos en el *estado del arte* descriptores que se acercan bastante al caso ideal planteado. Los más relevantes en este sentido son los descriptores SIFT [1], SURF [2], y más recientemente BRIEF [3]. El descriptor SIFT surgió del trabajo de David Lowe y, sin dudas, sentó un precedente sobre como resolver en forma satisfactoria el problema de reconocimiento de imágenes, y su descriptor SIFT fue ejemplo para muchos trabajos futuros. Este es el caso de SURF, donde sus autores proponen una suerte de reemplazo para SIFT, definiendo un descriptor mucho más rápido que éste, y en muchos casos obteniendo mejores resultados. Tal es así que hoy, el descriptor SURF, se ha convertido en el descriptor *estándar* a usar en este tipo de tareas.

El descriptor BRIEF publicado recientemente [3], nos interesa particularmente porque, con un enfoque novedoso, define un nuevo tipo de descriptor mucho más rápido que el propio SURF y de representación más compacta que éste, obteniendo resultados similares en muchos casos prácticos.



Estas características de velocidad y economía de representación, lo hacen especialmente atractivo para emplearse en aplicaciones que deban ejecutarse en tiempo real. Como se detallará más adelante, el trabajo de esta tesina se basa en el descriptor BRIEF.

No obstante el buen desempeño que tienen los descriptores mencionados, en un análisis más minucioso, no pasaría desapercibido el hecho de que todos ellos están diseñados sólo para trabajar con imágenes en formato de escala de grises. Esto no significa que no puedan aplicarse a imágenes a color, pero sí que las mismas deberán como primer paso cambiarse de formato, descartando toda información de color previamente existente. Sin embargo, una simple observación a las imágenes a las que accedemos habitualmente, bastará para comprobar que la mayoría, sino todas ellas, se capturan, se almacenan, y se visualizan en formatos conteniendo millones de colores, conteniendo potencialmente mucha más información que su correspondiente representación en escala de grises, contradiciendo en cierta medida la observación previa. Esta inquietud nos motivó a estudiar el tema y tratar de extender alguno de estos algoritmos conocidos, para que pueda ser aprovechada mejor parte de la información que habitualmente se desecha.

La contribución del presente trabajo es un nuevo descriptor local denominado `mcBrief` (multi-channel BRIEF), el cual surge de modificar el descriptor BRIEF [3] de manera de mejorar su efectividad cuando se trabaja con imágenes que poseen más de un canal de datos, como es el caso de las imágenes a color. Este nuevo descriptor caracteriza mejor los puntos presentes en imágenes multicanal, lo que se traduce en mejores resultados a la hora de encontrar correspondencias. Debido a que el descriptor `mcBrief` es de la misma longitud que el descriptor BRIEF original, y no se incrementa la complejidad de su extracción, esta mejora se obtiene sin sacrificar eficiencia y hace que conserve la cualidad de ser apto para ser usado en aplicaciones de tiempo real.

Esta tesina se organiza de la siguiente manera, en el Capítulo 2 se presenta la base teórica que introduce el vocabulario y los conceptos usados a lo largo del trabajo, en el Capítulo 3 se presenta un resumen de trabajos de otros autores que se relacionan con el nuestro porque han tratado de resolver el mismo problema o uno similar, en el Capítulo 4 se presenta la definición formal del descriptor `mcBrief` propuesto y se analizan en forma teórica sus propiedades, luego, en el Capítulo 5 se somete el nuevo descriptor a diferentes pruebas experimentales a fin de evaluar su desempeño y compararlo con el de otros descriptores, y finalmente, en el Capítulo 6 se resumen las diferentes conclusiones obtenidas en los capítulos anteriores, se presenta la conclusión general del trabajo y los temas pendientes de solución, o que requieren mayor análisis, y que se dejan para explorar en el futuro.

## Capítulo 2

# Conocimientos generales

A continuación, explicaremos algunos conceptos generales sobre las imágenes y su representación a modo de fijar un vocabulario común en lo que sigue de la tesina. También se expone la teoría sobre otros conceptos necesarios para una mejor comprensión de los capítulos posteriores.

### 2.1. Representación de imágenes

Las cámaras digitales capturan imágenes a color usando un lente para dirigir la luz reflejada en las superficies de los objetos que componen la escena sobre un sensor plano. Este sensor, de forma rectangular, está formado por múltiples sensores más pequeños que registran la cantidad de luz que cada uno de ellos recibe. Estos pequeños sensores no son todos iguales, sino que algunos miden únicamente la cantidad de luz roja (R), otros la cantidad de luz verde (G), y otros la cantidad de luz azul (B). Una cámara moderna suele tener millones de estos microsensores.

El proceso de captura de una imagen consiste puntualmente en registrar durante un tiempo muy breve la cantidad de luz que incide sobre el sensor de la cámara, y luego estimar la cantidad de luz de cada uno de los colores que correspondería a cada uno de los puntos de una grilla rectangular equidistante cubriendo la superficie del sensor. Cada uno de los puntos de la grilla, llamados *píxeles*, se almacenan en una matriz respetando su disposición espacial a la que se denomina *imagen digital* o simplemente *imagen*. En el caso descrito, cada píxel se representa mediante un vector de tres dimensiones, conteniendo en cada componente el valor estimado de la cantidad de luz de cada color correspondiente a esa posición. Una imagen codificada de esta manera se dice que contiene tres *canales* y que se encuentra en formato RGB.

El formato de imagen RGB, o espacio de color RGB, no es la única forma de codificar imágenes a color existente, aunque es el formato más común para el intercambio y almacenamiento de imágenes digitales. En algunos casos, resulta conveniente procesar las imágenes usando otros espacios de colores aunque, usualmente, la conversión se realiza sólo en memoria y convirtiéndose nuevamente al espacio de color RGB al finalizar el proceso, para su almacenamiento.

En muchas aplicaciones resulta irrelevante el color individual de cada píxel, importando sólo el total de luz incidente en cada punto. Cuando es así, es posible representar la imagen completa usando un único canal, donde cada píxel se representa mediante un único valor que mide su *brillo* o *intensidad*. Un imagen expresada en esta forma, se dice que está en formato de *escala de grises* y no puede volverse a convertir a un formato de color. Al contrario, una imagen en formato RGB puede convertirse fácilmente al formato de escala de grises por medio de la siguiente relación:

$$Gray_{x,y} = 0,30 R_{x,y} + 0,59 G_{x,y} + 0,11 B_{x,y} \quad (2.1)$$

## 2.2. Reconocimiento de imágenes

Una aplicación de *reconocimiento de imágenes* tiene el objetivo de determinar si dos imágenes dadas retratan a un mismo objeto o escena. Según el objetivo particular, en la literatura se menciona este mismo problema como de *reconocimiento de objetos* o de *reconocimiento de escenas*, según el tipo de imágenes con las que se trabaje. Para el ámbito de este trabajo no hace mayor diferencia si el objetivo específico es uno u otro, dado que el problema se resolverá de la misma forma, y hemos optado por llamar *reconocimiento de imágenes* al problema general de determinar la correspondencia entre imágenes.

Existen diferentes enfoques para la resolución de un problema de reconocimiento de imágenes, pero el que nos interesa se basa en el uso de descriptores locales para encontrar correspondencias entre pequeños sectores, llamados *patches*, de cada imagen. En este enfoque, el proceso se divide en al menos tres pasos bien diferenciados:

1. **Detección:** en esta etapa se aplica a una imagen dada un *algoritmo detector* que busca puntos que cumplan ciertas características. Según el algoritmo detector en uso, varía el tipo de puntos buscado, pero lo que se debe cumplir siempre es la característica de *repetibilidad*, es decir, lo que se desea es que aplicando el algoritmo a diferentes imágenes de la misma escena se obtengan aproximadamente los mismos puntos. Cada uno de los puntos detectados se denomina indistintamente *punto de interés*, *keypoint* o *feature*.
2. **Extracción de descriptores:** para cada uno de los puntos de interés detectados en la etapa anterior se genera una *firma* que lo identifique. La información contenida en la firma, como en el caso anterior, la determina el algoritmo aplicado. Esta firma es comúnmente llamada *descriptor local*, debido a que “describe” el patch de imagen correspondiente, y lo que se busca en este caso es la característica de *discriminabilidad* y la invariancia frente a los cambios que puedan afectar a la imagen. Discriminabilidad se refiere a que cada descriptor debe permitir, mediante comparación con otros descriptores, determinar si pertenece al mismo patch de imagen, o a patches distintos, con el mínimo error posible. La calificación de *local* se agrega para distinguir este tipo de descriptores de los *descriptores globales* que, en vez de identificar sólo un patch en la imagen, identifican a la imagen en su totalidad. En nuestro caso únicamente nos conciernen los descriptores locales, a los que muchas veces llamaremos sólo *descriptores*.
3. **Determinación de correspondencias (matching):** la última etapa es la búsqueda de un descriptor “correspondiente” para cada uno de los descriptores extraído anteriormente. Para esto se examina un conjunto de descriptores extraídos en forma previa desde la imagen o imágenes que forman la *base de datos* de búsqueda. Un par de descriptores formará una correspondencia, o *match*, cuando la distancia entre ellos sea mínima. El criterio exacto puede variar pero, en rasgos generales, la imagen almacenada en la base de datos que más descriptores en común tenga con la imagen de consulta, es la que se devuelve como resultado del problema de reconocimiento. Nótese que sólo basta con que una parte de los descriptores extraídos coincidan con los de la imagen en la base de datos para que el algoritmo tenga éxito, característica que hace al método tolerante a errores y robusto ante oclusiones parciales en las imágenes.

El foco de este trabajo es únicamente en la etapa de extracción de descriptores, aunque es conveniente tener presente todo el proceso para poder entender mejor el objetivo general e interpretar correctamente los resultados obtenidos.

## 2.3. Propiedades de un descriptor ideal

En el caso ideal, un descriptor local de features debería cumplir las siguientes propiedades:

- **Discriminabilidad:** debe existir muy baja probabilidad de detectar correspondencias incorrectas.
- **Eficiencia:** debe ser computacionalmente eficiente de extraer.
- **Invariancia a deformaciones habituales:** deben encontrarse correspondencias aún cuando estén presente en las imágenes los siguientes tipos de deformaciones:
  - ruido en la imagen
  - cambios en la iluminación de la escena
  - cambios de escala, rotación y otras deformaciones producidas por una variación de la posición de la cámara

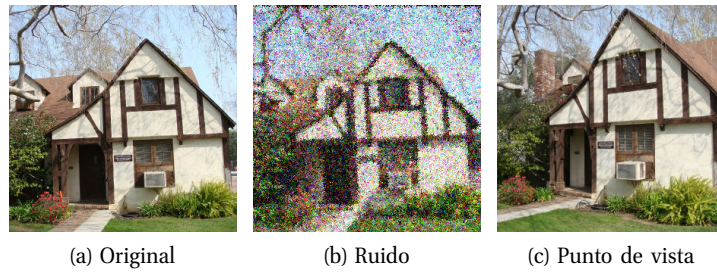


Figura 2.1: Muestra de imágenes con diferentes alteraciones

Aunque estos requisitos sólo se cumplirían completamente en el caso ideal, los descriptores que actualmente forman el *estado del arte* están muy cerca de dicho requisito. Habitualmente no se llegan a cumplir todos ellos porque los autores deben sacrificar algunos requerimientos en beneficio de otros, por ejemplo, es común sacrificar eficiencia para obtener mayor invariancia a deformaciones geométricas.

## 2.4. Modelo diagonal

El *modelo diagonal* [4, 5] permite modelar la relación entre imágenes de la misma escena tomadas bajo diferentes condiciones de iluminación. Este modelo será útil para estudiar como responde el descriptor `mcBrief`, que se presentará en el Capítulo 4, a los diferentes cambios de iluminación que se pueden encontrar en la práctica.

### Definición

Sean  $c$  y  $d$  dos condiciones de iluminación, y sean  $\mathbf{f}^c$  y  $\mathbf{f}^d$  las respectivas imágenes tomadas bajo  $c$  y  $d$ , entonces

$$\mathbf{f}^c = \mathcal{D}^{d,c} \mathbf{f}^d, \quad (2.2)$$

donde  $\mathcal{D}^{d,c}$  es una matriz diagonal que mapea los colores tomados en condiciones de iluminación  $d$  (desconocida), al valor que los mismos tendrían en condiciones de iluminación  $c$  (canónica). Si las imágenes se encuentran expresadas en el espacio de color RGB, la Ecuación 2.2 quedaría

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^d \\ G^d \\ B^d \end{pmatrix}. \quad (2.3)$$

El modelo diagonal no es suficiente para representar algunos casos de interés, por lo que posteriormente fue ampliado por Finlayson *et al.* [6] para incluir un *offset*  $(o_1, o_2, o_3)^T$ , dando lugar a lo que se conoce como el modelo *offset-diagonal*:

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^d \\ G^d \\ B^d \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}. \quad (2.4)$$

El modelo *offset-diagonal* nos permite representar cinco tipo de alteraciones que afectarán a las imágenes con las que trabajaremos, como consecuencia de diferencias en las condiciones de iluminación existentes al momento de su captura.

#### 2.4.1. Cambio de intensidad

Un *cambio de intensidad* ocurre cuando la intensidad de la luz varía por igual en todas las longitudes de onda. Visualmente, este cambio se percibe como una imagen más luminosa pero los colores individuales no se ven afectados, salvo en el caso en que el valor resultante exceda el límite de la representación subyacente. Un cambio de intensidad de factor  $a$  se expresa en el modelo diagonal como:

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R^d \\ G^d \\ B^d \end{pmatrix}. \quad (2.5)$$

La anterior ecuación, y el concepto de *cambio de intensidad*, además de modelar modificaciones en la intensidad de la fuente de luz, también contempla la presencia de sombras incoloras en la escena.

#### 2.4.2. Desplazamiento de intensidad

Un *desplazamiento de intensidad* equivalente en todos los canales de color debe ser expresado mediante el modelo *offset-diagonal*, su ecuación es la siguiente:

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} R^d \\ G^d \\ B^d \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}. \quad (2.6)$$

Este tipo de alteración se produce debido a efectos de luz de fondo difusa, por reflejos de luz blanca en objetos brillantes, y a causa de sensibilidad infraroja en los sensores de la cámara. Obsérvese que el valor de los colores individuales no se ve afectado.

#### 2.4.3. Cambio y desplazamiento de intensidad

El efecto de *cambio y desplazamiento de intensidad* es la acción combinada de las dos alteraciones anteriores, y se expresa mediante el modelo *offset-diagonal* así:

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R^d \\ G^d \\ B^d \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}. \quad (2.7)$$

#### 2.4.4. Cambio de color

La representación de un *cambio de color* se hace mediante la ecuación del modelo diagonal completo (Ecuación 2.3) sin ninguna restricción, es decir, se permite que  $a \neq b \neq c$ . El fenómeno de cambio de color está presente en todos los casos en que color percibido varía como consecuencia de un cambio en el color del iluminante de la escena, por ejemplo, al emplear una lámpara artificial, o por el efecto del cambio de color de la luz del sol a lo largo del día.

#### 2.4.5. Cambio y desplazamiento de color

Este último caso es el más general de todos los que se pueden expresar usando este modelo y se representa mediante el modelo offset-diagonal completo (Ecuación 2.4), incluyendo desplazamientos arbitrarios  $o_1 \neq o_2 \neq o_3$  y cambios de color  $a \neq b \neq c$ .

### 2.5. Transformaciones geométricas en 2D

A continuación se describen el conjunto básico de transformaciones que se pueden aplicar a una figura geométrica en el plano, para un ejemplo véase la Figura 2.2. Las mismas son de interés porque forman parte del conjunto de modificaciones que afectarán a los elementos en imágenes tomadas desde distintos puntos de vista. Para información más detallada acerca de este tópico y temas relacionados, puede consultarse la bibliografía de referencia [7].

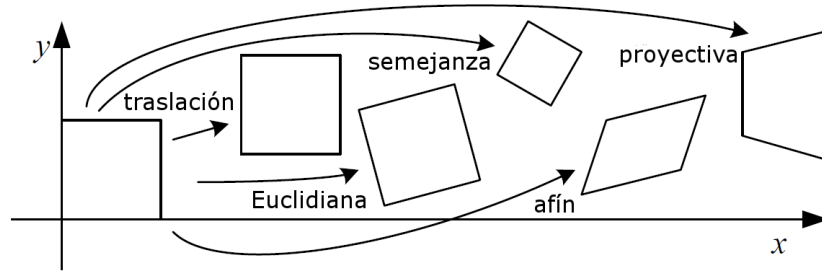


Figura 2.2: Transformaciones planares básicas

#### Puntos 2D

Los puntos 2D, que en nuestro caso representan coordenadas de píxeles en una imagen, se denotan usando un vector 2D,

$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^2. \quad (2.8)$$

Adicionalmente, cuando resulte conveniente, los puntos 2D serán representados mediante sus *coordenadas homogéneas* tal que  $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{w})^T \in \mathcal{P}^2$ , y  $\mathcal{P}^2 = \mathbb{R}^3 - (0, 0, 0)^T$  se denomina *espacio proyectivo*. En  $\mathcal{P}^2$ , todo par de vectores que difieran únicamente en su escala son considerados equivalentes. Un vector  $\tilde{\mathbf{x}}$ , en coordenadas homogéneas, puede convertirse en un vector  $\mathbf{x}$  de  $\mathbb{R}^2$  con sólo dividir por su último elemento  $\tilde{w}$ :

$$\tilde{\mathbf{x}} = \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{pmatrix} = \tilde{w} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \tilde{w} \bar{\mathbf{x}}, \quad (2.9)$$

donde  $\bar{\mathbf{x}} = (x, y, 1)^T$  es el *vector aumentado* de  $\mathbf{x} = (x, y)^T$ . Aquellos vectores en  $\mathcal{P}^2$  cuyo valor  $w$  es 0 se denominan *puntos de afinidad* y no tienen equivalente en coordenadas no homogéneas.

### 2.5.1. Traslación

Un vector  $\mathbf{x}$ , y una versión  $\mathbf{x}'$  del mismo que ha sido trasladada en el plano, se relacionan así:

$$\mathbf{x}' = \mathbf{x} + t, \quad \mathbf{x}' = \begin{pmatrix} I & t \end{pmatrix} \mathbf{x}. \quad (2.10)$$

La misma relación expresada mediante coordenadas homogéneas queda:

$$\tilde{\mathbf{x}}' = \begin{pmatrix} I & t \\ 0^T & 1 \end{pmatrix} \tilde{\mathbf{x}}. \quad (2.11)$$

Aunque la primera versión resulte más compacta, en general, se prefiere la versión expresada en coordenadas homogéneas porque, al tomar la forma de una matriz de  $3 \times 3$ , puede concatenarse mediante la operación de multiplicación con otros tipos de transformaciones.

### 2.5.2. Rotación + traslación

Este tipo de transformación también es conocida como *transformación Euclidiana 2D*, porque preserva las distancias Euclidianas entre elementos. Se expresa así:

$$\mathbf{x}' = R\mathbf{x} + t, \quad \mathbf{x}' = \begin{pmatrix} R & t \end{pmatrix} \mathbf{x}, \quad (2.12)$$

$$\tilde{\mathbf{x}}' = \begin{pmatrix} R & t \\ 0^T & 1 \end{pmatrix} \tilde{\mathbf{x}}, \quad (2.13)$$

donde  $R = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}$  cumple que  $RR^T = I$  y  $|R| = 1$ , es decir,  $R$  es una matriz ortonormal de rotación.

La ecuación anterior se puede modificar, agregándole un factor de escala  $s$ , para hacerla más general y poder representar también una *relación de semejanza*. La ecuación modificada quedaría así:

$$\mathbf{x}' = sR\mathbf{x} + t, \quad \mathbf{x}' = \begin{pmatrix} sR & t \end{pmatrix} \mathbf{x}. \quad (2.14)$$

### 2.5.3. Transformación afín

La *transformación afín* se obtiene permitiendo en la ecuación  $\mathbf{x}' = A\mathbf{x}$  que la matriz  $A \in \mathbb{R}^{2 \times 3}$  varíe arbitrariamente.

$$\mathbf{x}' = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix} \mathbf{x}. \quad (2.15)$$

Una transformación afín se caracteriza porque preserva el paralelismo entre rectas.

### 2.5.4. Transformación proyectiva

Las *transformaciones proyectivas*, como su nombre lo indica, operan en un *espacio proyectivo*, por lo que los vectores se deben expresar en *coordenadas homogéneas*,

$$\tilde{\mathbf{x}}' = \tilde{H}\tilde{\mathbf{x}}, \quad (2.16)$$

donde  $\tilde{H} \in \mathbb{R}^{3 \times 3}$  es una *matriz homogénea* arbitraria. Las matrices homogéneas, al igual que los vectores análogos, están definidas únicamente hasta su escala, significando que son equivalentes entre sí todas las matrices que difieran únicamente en su escala. El vector  $\tilde{\mathbf{x}}'$  obtenido al aplicar este tipo de

transformación, deberá ser normalizado si lo que se desea es obtener su equivalente  $\mathbf{x}'$  en coordenadas no homogéneas:

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \quad y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}. \quad (2.17)$$

Las transformaciones proyectivas se conocen también con el nombre de *homografía* y también como *transformación de perspectiva*. Las transformaciones proyectivas tienen la característica de preservar la rectitud de las líneas, es decir, las líneas rectas permanecen rectas luego de transformadas.

### 2.5.5. Jerarquía de transformaciones 2D

La Tabla 2.1 resume todas las transformaciones 2D vistas hasta aquí. La forma más simple de interpretarlas es pensando en cada una de ellas como una matriz (posiblemente restringida) de  $3 \times 3$  expresada en coordenadas homogéneas. Aquellas transformaciones expresadas como matrices de  $2 \times 3$ , pueden ser extendidas añadiéndoles la fila  $(0^T \mid 1)$  para convertirlas en homogéneas.






Transformación	Matriz	Preserva	# G.L. <sup>(*)</sup>	Figura
Traslación	$\left( \begin{array}{c c} I & t \\ \hline & \end{array} \right)_{2 \times 3}$	orientación	2	
Euclidiana: traslación y rotación	$\left( \begin{array}{c c} R & t \\ \hline & \end{array} \right)_{2 \times 3}$	longitud	3	
Semejanza: traslación, rotación y escala	$\left( \begin{array}{c c} sR & t \\ \hline & \end{array} \right)_{2 \times 3}$	ángulos	4	
Afin	$\left( \begin{array}{c} A \\ \hline \end{array} \right)_{2 \times 3}$	paralelismo	6	
Proyectiva	$\left( \begin{array}{c} \tilde{H} \\ \hline \end{array} \right)_{3 \times 3}$	rectitud	8	

Tabla 2.1: Jerarquía de transformaciones 2D  
(\*)G.L.: Grados de Libertad

## 2.6. Imagen integral

El concepto de imagen integral [8] se refiere a un algoritmo que permite calcular la suma de los valores, en una sección rectangular cualquiera de una imagen, en forma rápida y en tiempo constante independientemente del tamaño de la sección. Esto se consigue construyendo, en forma anticipada, una tabla conteniendo en cada celda la suma acumulada de todos los valores de las columnas y filas anteriores, véase la Figura 2.3.

Dada una matriz  $\mathcal{I} \in \mathbb{R}^{n \times m}$ , con  $n$  filas y  $m$  columnas, su imagen integral asociada es una matriz  $I \in \mathbb{R}^{n \times m}$  tal que

$$I(x, y) = \sum_{i \leq x, j \leq y} \mathcal{I}(i, j). \quad (2.18)$$

La construcción de  $I$  se realiza en forma eficiente, y recorriendo una única vez la imagen, aprovechando la propiedad de que

$$I(x, y) = \mathcal{I}(x, y) + I(x - 1, y) + I(x, y - 1) - I(x - 1, y - 1). \quad (2.19)$$



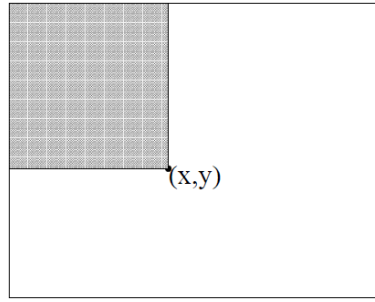


Figura 2.3: en la *imagen integral* el valor en cada punto  $(x, y)$  es la suma de todas las de las columnas y filas anteriores de la imagen original.

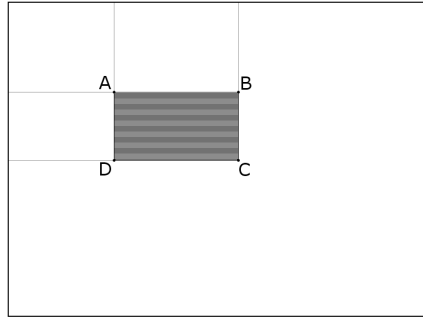


Figura 2.4: usando la *imagen integral* la suma de todos los puntos en un área rectangular de  $\mathcal{I}$  se obtiene mediante 3 operaciones aritméticas,  $I(A) + I(C) - I(B) - I(D)$ .

Luego, el cálculo de la suma de todos los valores en una sección rectangular  $\overline{ABCD}$  en  $\mathcal{I}$ , como la de la Figura 2.4, se obtiene por medio de 4 operaciones de acceso a la imagen integral y 3 operaciones de suma y resta:

$$\sum_{(x,y) \in \overline{ABCD}} \mathcal{I}(x, y) = I(A) + I(C) - I(B) - I(D). \quad (2.20)$$

El uso de imágenes integrales permite acelerar considerablemente diferentes algoritmos que necesitan calcular la suma de sectores rectangulares de imagen, en múltiples ubicaciones y en tamaños diferentes. Este es el caso del algoritmo que extrae descriptores `mcBrief`, definido en el Capítulo 4, donde el costo inicial de construir la imagen integral se amortiza a medida que la cantidad de descriptores a extraer crece.

## 2.7. Evaluación de algoritmos de matching

Dado un algoritmo de matching, se puede cuantificar su desempeño contando el número de resultados correctos e incorrectos a partir de las siguientes definiciones:

- **TP:** *True Positives*, número de matches detectados correctamente.
- **FP:** *False Positives*, número de matches incorrectos devueltos como correctos.
- **FN:** *False Negatives*, número de matches correctos no detectados.
- **TN:** *True Negatives*, número de resultados incorrectos.

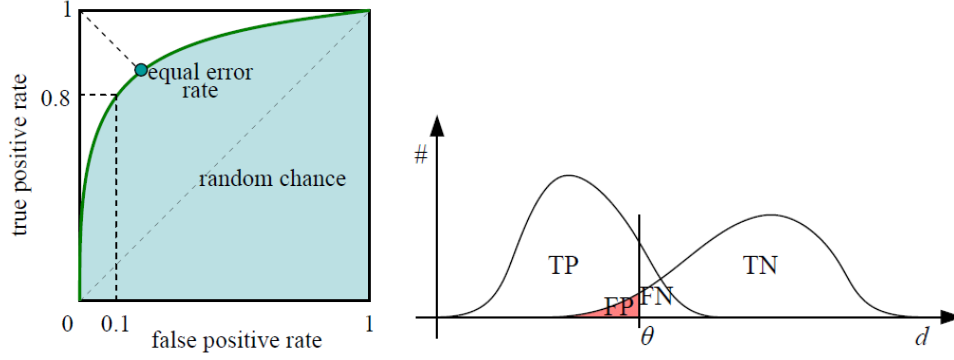


Figura 2.5: en (a) se muestra como ejemplo la curva ROC que se obtendría a partir de calcular los valores de FPR y TPR variando el valor de  $\theta$  en (b).

Usando estos cuatro conceptos se pueden definir varias métricas útiles para poder comparar diferentes algoritmos:

$$TPR (recall) = \frac{TP}{TP + FN} = \frac{TP}{P} \quad (2.21)$$

$$FPR = \frac{FP}{FP + TN} = \frac{FP}{N} \quad (2.22)$$

$$precision = \frac{TP}{TP + FP} \quad (2.23)$$

El valor de *True Positive Rate* (TPR), también conocido como *recall* o *exhaustividad* en el área de recuperación de información, mide cuantos resultados correctos han sido devueltos de todos los resultados correctos posibles. En cambio, el *False Positive Rate* (FPR) mide cuantos resultados incorrectos han sido devueltos del total de resultados incorrectos existentes. Por último, la *precisión* mide cuantos resultados devueltos como correctos son realmente correctos.

Una estrategia o algoritmo de matching, usando un parámetro o threshold en particular, puede ser caracterizado mediante sus valores de TPR y FPR, en el caso ideal tendríamos  $TPR = 1$  y  $FPR = 0$ . A medida que se varía el valor de threshold, se obtendrá una familia de valores TPR y FPR, que habitualmente se conoce como curva ROC (Receiver Operating Characteristic) [9], Figura 2.5. En el caso de querer comparar diferentes algoritmos, o diferentes configuraciones, es útil graficar la curva ROC correspondiente a cada uno de ellos y aquella curva que se aproxime más al punto ideal (0, 1) corresponderá al mejor caso. Además, es posible calcular el área cubierta por la curva y obtener así un único escalar que cuantifica el desempeño del algoritmo, este valor se conoce como AUC (Area Under the Curve). Otro valor escalar comúnmente usado para medir el desempeño es el *Equal Error Rate* (EER) que corresponde al punto en que  $(FPR = 1 - TPR)$ .

En forma análoga a como se obtiene la curva ROC, es posible calcular el valor de *recall* y *precisión*, y usar este par, en reemplazo del par (FPR, TPR), para caracterizar los diferentes algoritmos de matching. Algunos autores [10, 2] prefieren graficar el valor de  $1 - precision$  versus el valor de *recall*, otros [4] prefieren graficar *recall* versus *precisión*. En todos los casos, los resultados son similares, pero la interpretación de las gráficas variará. En este trabajo, usaremos las curvas  $1 - precision$  vs. *recall* para evaluar los resultados en forma gráfica, y generaremos las curvas ROC para calcular los valores de AUC y ERR que nos permitirán una comparación numérica.

## Capítulo 3

# Trabajos relacionados

### 3.1. SIFT: Scale-Invariant Features

La referencia obligatoria cuando se discute sobre descriptores locales es indudablemente el trabajo de David Lowe [1]. En el mismo, se brinda una solución para la etapa de detección, para la creación de descriptores y para la etapa de matching, constituyendo una solución completa y relativamente rápida para la época. El detector propuesto por Lowe es un filtro por Diferencia de Gaussianas (DoG), que es una aproximación del filtro basado en el Laplaciano de Gaussianas (LoG) presentado por Mikolajczyk y Schmid [11], más rápida de calcular. Con el propósito de obtener invariancia a cambios de escala, lo que se hace es construir una pirámide de imágenes donde en cada nivel se aplica un filtro ‘mas grueso’, se aumenta el valor de  $\sigma$  de la función Gaussiana, de modo de barrer todas las escalas. De este modo, se obtiene de cada punto de interés, junto con sus coordenadas, la escala en la que fue detectado lo que permite generar el descriptor en la escala óptima. El descriptor que se usa, denominado SIFT, toma la forma de un vector de 128 dimensiones que codifica valores del gradiente de brillo, en el entorno al punto de interés, cuantificados mediante histogramas. Este descriptor ha probado ser altamente discriminante a la hora de determinar correspondencias y, al tomar la forma de un vector, resulta apropiado emplear la distancia Euclidiana para su comparación.

Hoy en día se considera que, si bien cumple su objetivo adecuadamente, SIFT es ‘un poco lento’ a la hora de aplicarse. Esta lentitud se debe principalmente a que como primer paso se debe construir la pirámide con una imagen para cada una de las escalas que se desea considerar, y a que la comparación entre sus descriptores de 128 dimensiones lleva un tiempo no despreciable.

### 3.2. SURF: Speeded Up Robust Features

Siguiendo las ideas de SIFT, Bay *et al.* [2] presentan un nuevo detector y descriptor, denominado SURF, el cual es más rápido de calcular y evaluar que su predecesor. En este caso, el detector denominado ‘Fast-Hessian’ aproxima el determinante de la matriz Hessiana [11] por medio de un ‘box-filter’ (filtro de cajas) y el descriptor es un vector de 64 componentes que se construye mediante una aproximación de las repuestas a una ‘Haar-wavelet’. La ventaja de ambas aproximaciones radica en que se pueden calcular en unas pocas operaciones aprovechando las propiedades de las imágenes integrales [8]. Además, para la realizar la detección en múltiples escalas basta con variar el tamaño del filtro usado por el detector, y de las wavelets al generar el descriptor, reusando la imagen integral calculada previamente, y omitiendo completamente la construcción de la pirámide de imágenes que requiere SIFT. Para la comparación entre descriptores SURF, al igual que en el caso anterior, se emplea la distancia Euclidiana.

Las aproximaciones propuestas han probado ser efectivas, logrando acelerar notablemente el proceso de detección de puntos de interés y generación de descriptores sin pérdida de efectividad. La etapa de matching también es considerablemente más rápida, debido a que el descriptor SURF posee la mitad

del tamaño que su correspondiente en SIFT. Estas características hacen que SURF se haya convertido casi en un estándar *de facto* para muchas aplicaciones de *Computer Vision*.

En el mismo trabajo [2] también se propone un descriptor alternativo, denominado U-SURF (Up-Right SURF), que puede ser usado cuando se requiere mayor velocidad, y especialmente si no se requiere corrección de orientación, ya que la ganancia de velocidad se logra a costa de omitir este paso.

### 3.3. BRIEF: Binary Robust Independent Elementary Features

A pesar de la mejora en velocidad que representó SURF respecto de SIFT, el proceso de detección y extracción de descriptores sigue siendo relativamente complejo, y su descriptor de 64 dimensiones aún requiere un tiempo no despreciable para su comparación. Estas razones causan que en general SURF no sea apropiado para procesamiento en tiempo real de secuencias de vídeo, requerimiento habitual en una aplicación de realidad aumentada o de vSLAM entre otras. Para satisfacer esta necesidad se deben emplear conceptos más simples y actualmente existen varios proyectos adhieren a esta tendencia. Así, han surgido una serie de algoritmos detectores y descriptores “rápidos”, como alternativas a SURF y SIFT, para emplear en aplicaciones con estrictos requerimientos temporales. Ejemplos de ellos son FAST [12], FASTER [13], CenSurE [14], y SUSurE [15]. No vamos a entrar en detalles de como opera cada uno de ellos porque escapa al objetivo de este trabajo, pero el lector interesado puede consultar la bibliografía.

Recientemente, Calonder *et al.* [3] ha presentado un descriptor, que adhiriendo a este concepto de búsqueda de simplicidad en pos de velocidad, emplea una idea novedosa para acelerar el proceso de extracción y también la etapa de detección. Este descriptor, denominado BRIEF, está constituido por una cadena de bits, a diferencia de SIFT y SURF que representan vectores, la cual se genera como resultado de un conjunto de tests binarios muy simples calculados a partir de una imagen integral [8]. Esta característica, hace que resulte apropiado el uso de la distancia de Hamming para determinar la similitud entre descriptores, lo cual resulta particularmente conveniente porque la distancia de Hamming es muy simple de calcular, hasta pudiéndose realizar mediante una única instrucción de máquina en algunos procesadores modernos<sup>1</sup>, y definitivamente más simple que el cálculo de la distancia Euclidiana entre vectores de dimensiones comparables. Una ventaja adicional es que la práctica demuestra que, generalmente, un descriptor BRIEF de 256 bits (32 bytes) es suficiente para lograr resultados similares a los que se obtienen mediante un descriptor SURF estándar de 64 dimensiones (256 bytes).

A partir del párrafo anterior, se desprende que los descriptores BRIEF cumplen varias características deseables, son simples de generar, pueden compararse en forma eficiente, y su representación es extremadamente concisa, economizando memoria y acelerando el proceso de matching. Estas razones motivaron que haya sido el elegido como punto de partida de este trabajo y constituya la base del descriptor *mcBrief* detallado más adelante.

### 3.4. Descriptores Color

Los descriptores mencionados hasta aquí, todos han sido diseñados para aplicarse a imágenes en escala de grises. Debido al éxito de los mismos en este ámbito, y sumado a la creciente disponibilidad de imágenes capturadas y almacenadas en color, han habido varios intentos de extender las ideas existentes a este nuevo dominio de aplicación. Según se expone en [4], los intentos de crear descriptores color se dividen en tres grupos: descriptores basados en histogramas de color, descriptores basados en momentos generalizados de color, y descriptores basados en SIFT.

Los descriptores locales basados en histogramas, en general, todos aplican la idea de construir un histograma para cada uno de los canales presentes en la imagen color y concatenarlos formando un descriptor compuesto. Así, la diferencia entre un descriptor y otro, en general, consiste en el número

<sup>1</sup>Procesadores que implementen la instrucción POPCNT, o similar, detallada en la especificación de SSE4.2[16]

de *bins* y dimensiones de cada histograma, y del espacio de color en el que se exprese la imagen. En [4] se mencionan en esta categoría los descriptores: *RGB histogram*, *Opponent histogram*, *Hue histogram*, *rg-histogram*, y *Transformed color distribution*.

Considerando a una imagen color en formato RGB como una función

$$I : (x, y) \mapsto (R(x, y), G(x, y), B(x, y))$$

es posible aplicar el concepto matemático de momentos a ella. Esta idea ha sido tomada por Mindru *et al.* [17] dando lugar a lo que se conoce como *momentos generalizados de color*, existiendo 27 de ellos si se consideran los de primer y segundo orden. Ha sido propuesto crear un vector de 27 dimensiones conteniendo los valores de los momentos y usarlo como un descriptor color, y también crear un descriptor con 24 dimensiones conteniendo un conjunto de invariantes [17] definidos a partir de los mismos. Ambos son ejemplos de los descriptores basados en momentos generalizados de color que mencionan van de Sande *et al.* en [4].

El grupo de descriptores basados en SIFT, similarmente a los que se basan en histogramas, consiste en crear descriptores compuestos a partir de la concatenación de descriptores SIFT extraídos de los canales individuales de color. En este grupo se encuentran los descriptores *HSV-SIFT* [18], *HueSIFT* [19], *OpponentSIFT* [4], y *C-SIFT* [20].

En resumen, cuando de extraer descriptores locales de imágenes a color se trata, los autores en su mayoría optan por elegir un descriptor que se sepa da buenos resultados en imágenes de un único canal, y extraer un descriptor de este tipo de cada uno de los canales de color, formando un descriptor compuesto. Según lo expuesto anteriormente, es común aplicar esta estrategia usando histogramas y descriptores SIFT, pero naturalmente también se podrían usar descriptores SURF o BRIEF con resultados similares. Sin embargo, mediante este proceso usualmente se obtienen descriptores tres veces más grandes que los descriptores originales, triplicando los requerimientos de espacio y ralentizando el proceso de extracción y matching, sin que los beneficios así obtenidos sean también igual de generosos. Por esta razón, nosotros creemos que esta forma de proceder no es óptima y en este trabajo hemos preferido un enfoque diferente.

## Capítulo 4

# Descriptor mcBrief

El descriptor `mcBrief`, el cual constituye la propuesta de esta tesina, es una extensión al descriptor BRIEF [3] que, a diferencia de éste, puede ser aplicado en forma directa a imágenes color o cualquier otro tipo de imagen multicanal sin necesidad de convertirla previamente a un formato de un único canal, usualmente el formato de escala de grises. En la siguiente sección se explica en detalle este nuevo descriptor. Luego, se analiza la complejidad algorítmica de su extracción, y finalmente se analiza su comportamiento ante cada uno de los invariantes vistos.

### 4.1. Descripción

El descriptor `mcBrief` se construye concatenando el resultado de una serie de tests binarios muy simples que se evalúan en el entorno del punto de interés. De este modo, `mcBrief` toma la forma de una cadena de bits, siendo cada bit el resultado de uno de los tests, cuya longitud se puede variar de forma arbitraria ajustando la cantidad de tests a evaluar. Siguiendo la notación introducida en [3], identificaremos a los descriptores según su longitud en bytes siendo `mcBrief-16`, `mcBrief-32` y `mcBrief-64` descriptores de 128 bits, 256 bits y 512 bits respectivamente.

#### 4.1.1. Definición formal

Sea  $\mathcal{I}$  una imagen con  $n$  canales de datos, sea  $\mathbf{p}$  un patch en  $\mathcal{I}$  de tamaño  $S \times S$ , luego un test  $\tau$  en  $\mathbf{p}$  se define como:

$$\tau(\mathbf{p}; x, y, \alpha) = \begin{cases} 1 & \text{si } \alpha^T \mathbf{p}(x) < \alpha^T \mathbf{p}(y) \\ 0 & \text{sino.} \end{cases}, \quad (4.1)$$

donde  $x, y \in \mathbb{R}^2$  son dos ubicaciones;  $\mathbf{p}(x), \mathbf{p}(y) \in \mathbb{R}^n$  son vectores con los valores de los píxeles  $x$  e  $y$  en una versión suavizada de  $\mathbf{p}$ ; y  $\alpha \in \mathbb{R}^n$  es un vector tal que  $\alpha \sim N[\mu, \sigma^2]$ . Entonces, cada conjunto  $\{(x_1, y_1, \alpha_1), \dots, (x_k, y_k, \alpha_k)\}$  de ubicaciones  $(x, y)$  y coeficientes  $\alpha$  define unívocamente un conjunto de  $k$ -tests binarios sobre el patch  $\mathbf{p}$ .

Dado un conjunto de  $k$ -tests binarios, el descriptor `mcBrief` es la cadena de  $k$ -bits

$$f_k(\mathbf{p}) := \sum_{1 \leq i \leq k} 2^{i-1} \tau(\mathbf{p}; x_i, y_i, \alpha_i). \quad (4.2)$$

En forma experimental se ha determinado que los valores  $(x, y, \alpha)$  que generan mejores resultados son

$$(x, y, \alpha) \sim \left( N \left[ 0, \frac{1}{25} S^2 \right], N \left[ 0, \frac{1}{25} S^2 \right], N[0, 1] \right), \quad (4.3)$$

siendo esta la configuración usada durante el resto del trabajo. Nótese que tanto  $x$ ,  $y$ , como  $\alpha$ , son vectores y las distribuciones normales de probabilidad de la que son muestreados, son distribuciones isotrópicas. La Figura 4.1 muestra un ejemplo de la disposición espacial que toman el conjunto de tests binarios que componen un descriptor.

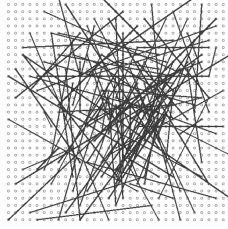


Figura 4.1: Ejemplo de disposición espacial de los tests binarios sobre un patch de imagen.

Dado que `mcBrief` se define como una cadena binaria, resulta apropiado el uso de la distancia de Hamming como medida de similitud entre descriptores. Emplear la distancia de Hamming es particularmente conveniente, porque es muy simple de calcular, hasta pudiéndose realizar mediante una única instrucción de máquina en algunos procesadores modernos<sup>1</sup>, y definitivamente más simple que la comparación mediante la distancia Euclidiana que requieren otros descriptores como SIFT y SURF.

## 4.2. Complejidad

Dada una imagen con un total de  $N$  píxeles y  $t$  canales, para extraer  $n$  descriptores `mcBrief` de  $k$ -bits de longitud de ella se deben completar los siguientes pasos, cada uno con la complejidad algorítmica descrita:

1. *Construir una imagen integral:*  $O(t \times N)$ , se debe recorrer toda la imagen y en cada píxel se almacena la suma acumulada. La complejidad de cada suma individual no se considera.
2. *Evaluar cada test binario:*  $O(2 \times t \times p)$ , se deben evaluar ambos miembros del test binario donde para cada canal se calcula el valor que tendría el píxel correspondiente en una versión suavizada del patch. Este valor se estima por medio de cuatro operaciones de suma, usando la imagen integral,  $p$  representa el costo de este cálculo.
3. *Extraer un descriptor:*  $O(k \times 2tp)$ , se deben evaluar  $k$  tests binarios por cada descriptor.
4. *Extraer  $n$  descriptores:*  $O(n \times 2tkp)$

La complejidad total partiendo de la imagen original, construyendo la imagen integral necesaria para calcular tests, y extraer los  $n$  descriptores, es  $O(tN + n \times 2tkp)$ . Una imagen color típica de  $800 \times 600$  tiene un total de  $N = 480000$  píxeles y  $t = 3$  canales. Un descriptor típico `mcBrief-32` tiene una longitud  $k = 256$  bits y el valor de  $p$  es el costo de calcular unas pocas operaciones de suma. En una situación como la descrita, la complejidad algorítmica de la extracción de descriptores `mcBrief` es aproximadamente lineal con respecto al número de píxeles:

$$O(tN + n \times 2tkp) \approx O(N) \quad (4.4)$$

Este resultado teórico, sumado al hecho de que la búsqueda de correspondencias se realiza por medio de la distancia de Hamming, indica que `mcBrief` puede ser implementado en forma eficiente, característica fundamental si se pretende usarlo en aplicaciones de tiempo real.

<sup>1</sup>Procesadores que implementen la instrucción `POPCNT`, o similar, detallada en la especificación de `SSE4.2`[16]

### 4.3. Invariantes

#### 4.3.1. Ruido en la imagen y deformaciones afines

La presencia de ruido en la imagen se verá como píxeles que poseen un valor diferente al que deberían tener si la imagen estuviera libre de ruido. El mismo puede aparecer en la imagen por razones diversas, incluyendo desde defectos en el sensor de la cámara hasta artificios introducidos por algoritmos de compresión aplicados a la imagen. El ruido puede que afecte uno o más canales simultáneamente.

El descriptor `mcBrief` obtiene cierta resistencia al ruido debido al hecho de que cada uno de los tests no se calcula sobre píxeles individuales, sino sobre un promedio, estimado por medio de la imagen integral, de todos los píxeles en el entorno a cada una de las coordenadas involucradas. De esta forma, si el ruido no es excesivo, la alteración de uno o más píxeles no afecta el resultado del test. Además, cada descriptor se obtiene a partir de una gran número de tests, donde cada uno sólo contribuye un bit, por lo tanto, aunque el ruido presente sea tal que afecte el resultado de alguno de los tests, sólo logrará alterar unos pocos bits en el descriptor final. Obsérvese también, que todos los bits tienen la misma importancia dentro del descriptor y la perturbación de alguno de ellos no afecta demasiado la medida de distancia. En resumen, ante la presencia de poco ruido el descriptor no se ve afectado, y en caso de ruido moderado sólo se perturbarán unos pocos bits del mismo sin impedir determinar la correspondencia correcta. Si el ruido fuera excesivo, el desempeño se verá medianamente afectado.

Recordemos que las transformaciones afines son aquellas que preservan el paralelismo de las líneas rectas. Cuando un patch de imagen, originalmente cuadrado, se ve afectado por este tipo de transformación se presentará como un paralelogramo. Excepto que el detector empleado detecte esta situación, como es el caso de detectores afines, y la misma sea tenida en cuenta, la región deformada será muestreada como si en realidad no lo estuviera, causando que cada test no se evalúe en la coordenada correspondiente. En caso de una diferencia de unos pocos píxeles, la misma se compensará, al igual que en el caso anterior, debido que los tests no consideran el valor del píxel en la coordenada exacta, sino que realizan un promedio en torno a esta. A medida que la diferencia se haga más pronunciada, los tests se comenzarán a ver afectados degradando el resultado proporcionalmente, hasta el caso extremo de impedir encontrar la correspondencia correcta.

En conclusión, el descriptor `mcBrief` no es invariante al ruido ni a las deformaciones afines, pero tolera ambas alteraciones en situaciones promedio, sin que logren degradar su desempeño general.

#### 4.3.2. Cambios en la iluminación de la escena

El análisis de como se adapta el descriptor `mcBrief` a los diversos cambios de iluminación posibles lo haremos por medio del modelo offset-diagonal presentado con anterioridad. Para el propósito de este análisis sólo se considerará el caso en que las imágenes se encuentran en formato RGB. El análisis de otros formatos y otros espacios de colores requeriría probablemente un modelo diferente para cada uno de los casos y escapa del alcance de este trabajo.

##### Cambio de intensidad

Sea  $\mathbf{p}^c$  un patch de imagen tomado bajo el iluminante canónico y sea  $\mathbf{p}^d$  el mismo patch pero alterado por un cambio de intensidad, según el modelo diagonal ambos patches se relacionan así

$$\mathbf{p}^c(x) = a\mathbf{p}^d(x). \quad (4.5)$$

Entonces, el resultado de evaluar un test  $\tau(\mathbf{p}^c; x, y, \alpha)$  será 1 si y sólo si  $\alpha^T \mathbf{p}^c(x) < \alpha^T \mathbf{p}^c(y)$ , reemplazando con la Ecuación 4.5 queda



$$\alpha^T \mathbf{p}^c(x) < \alpha^T \mathbf{p}^c(y) \Leftrightarrow \alpha^T a\mathbf{p}^d(x) < \alpha^T a\mathbf{p}^d(y) \Leftrightarrow \alpha^T \mathbf{p}^d(x) < \alpha^T \mathbf{p}^d(y),$$

$$\tau(\mathbf{p}^c; x, y, \alpha) = 1 \Leftrightarrow \tau(a\mathbf{p}^d; x, y, \alpha) = 1 \Leftrightarrow \tau(\mathbf{p}^d; x, y, \alpha) = 1. \quad (4.6)$$

Por lo tanto, un descriptor `mcBRIEF` extraído del patch  $\mathbf{p}^c$  y uno extraído del patch  $\mathbf{p}^d$  serán iguales bit a bit, cumpliendo la característica de ser invariante a un cambio de intensidad.

### Desplazamiento de intensidad

Siguiendo un análisis análogo al anterior y usando la misma notación, ante un caso de desplazamiento de intensidad, los patches  $\mathbf{p}^c$  y  $\mathbf{p}^d$  se relacionarán de la siguiente manera

$$O_1 = \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix},$$

$$\mathbf{p}^c(x) = \mathbf{p}^d(x) + O_1. \quad (4.7)$$

Luego, un test  $\tau(\mathbf{p}^c; x, y, \alpha)$  será 1 si y sólo si  $\alpha^T \mathbf{p}^c(x) < \alpha^T \mathbf{p}^c(y)$ , reemplazando con la Ecuación 4.7 tenemos

$$\alpha^T \mathbf{p}^c(x) < \alpha^T \mathbf{p}^c(y) \Leftrightarrow \alpha^T [\mathbf{p}^d(x) + O_1] < \alpha^T [\mathbf{p}^d(y) + O_1]$$

$$\Leftrightarrow \alpha^T \mathbf{p}^d(x) + (\alpha^T O_1) < \alpha^T \mathbf{p}^d(y) + (\alpha^T O_1) \Leftrightarrow \alpha^T \mathbf{p}^d(x) < \alpha^T \mathbf{p}^d(y),$$

$$\tau(\mathbf{p}^c; x, y, \alpha) = 1 \Leftrightarrow \tau(\mathbf{p}^d + O_1; x, y, \alpha) = 1 \Leftrightarrow \tau(\mathbf{p}^d; x, y, \alpha) = 1. \quad (4.8)$$

Queda así demostrado que el descriptor `mcBRIEF` es invariante a un desplazamiento de intensidad.

### Cambio y desplazamiento de intensidad

En el caso en que un cambio y un desplazamiento de intensidad ocurran en forma simultánea

$$\mathbf{p}^c(x) = a\mathbf{p}^d(x) + O_1, \quad (4.9)$$

$$\alpha^T \mathbf{p}^c(x) < \alpha^T \mathbf{p}^c(y) \Leftrightarrow \alpha^T [a\mathbf{p}^d(x) + O_1] < \alpha^T [a\mathbf{p}^d(y) + O_1]$$

$$\Leftrightarrow \alpha^T a\mathbf{p}^d(x) + (\alpha^T O_1) < \alpha^T a\mathbf{p}^d(y) + (\alpha^T O_1) \Leftrightarrow \alpha^T \mathbf{p}^d(x) < \alpha^T \mathbf{p}^d(y),$$

$$\tau(\mathbf{p}^c; x, y, \alpha) = 1 \Leftrightarrow \tau(a\mathbf{p}^d + O_1; x, y, \alpha) = 1 \Leftrightarrow \tau(\mathbf{p}^d; x, y, \alpha) = 1. \quad (4.10)$$

De esta manera, el descriptor también es invariante a cambios y desplazamientos simultáneos de intensidad en la imagen.

### Cambio y desplazamiento de color

Un cambio y desplazamiento de color requiere usar el modelo offset-diagonal completo y permitir que la matriz  $\mathcal{D}^{d,c}$  varíe libremente

$$\begin{aligned}\mathbf{p}^c(x) &= \begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix}, \quad \mathbf{p}^d(x) = \begin{pmatrix} R^d \\ G^d \\ B^d \end{pmatrix}, \\ \mathbf{p}^c(x) &= \mathcal{D}^{d,c} \mathbf{p}^d(x) + O = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^d \\ G^d \\ B^d \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}, \\ \mathbf{p}^c(x) &= \begin{pmatrix} aR^d + o_1 \\ bG^d + o_2 \\ cB^d + o_3 \end{pmatrix}.\end{aligned}\tag{4.11}$$

Reemplazando, como en los casos anteriores, la Ecuación 4.11 en el cálculo de los tests binarios tenemos

$$\begin{aligned}\alpha^T \mathbf{p}^c(x) < \alpha^T \mathbf{p}^c(y) &\Leftrightarrow \alpha^T \begin{pmatrix} aR^d(x) + o_1 \\ bG^d(x) + o_2 \\ cB^d(x) + o_3 \end{pmatrix} < \alpha^T \begin{pmatrix} aR^d(y) + o_1 \\ bG^d(y) + o_2 \\ cB^d(y) + o_3 \end{pmatrix} \\ &\Leftrightarrow \alpha_1[aR^d(x) + o_1] + \alpha_2[bG^d(x) + o_2] + \alpha_3[cB^d(x) + o_3] \\ &\quad < \alpha_1[aR^d(y) + o_1] + \alpha_2[bG^d(y) + o_2] + \alpha_3[cB^d(y) + o_3] \\ &\Leftrightarrow \alpha_1 aR^d(x) + \alpha_2 bG^d(x) + \alpha_3 cB^d(x) + [\alpha_1 o_1 + \alpha_2 o_2 + \alpha_3 o_3] \\ &\quad < \alpha_1 aR^d(y) + \alpha_2 bG^d(y) + \alpha_3 cB^d(y) + [\alpha_1 o_1 + \alpha_2 o_2 + \alpha_3 o_3] \\ &\Leftrightarrow \alpha^T \mathbf{p}^d(x) < \alpha^T \mathbf{p}^d(y), \\ \tau(\mathbf{p}^c; x, y, \alpha) &= 1 \Leftrightarrow \tau(\mathcal{D}^{d,c} \mathbf{p}^d + O; x, y, \alpha) = 1 \Leftrightarrow \tau(\mathbf{p}^d; x, y, \alpha) = 1.\end{aligned}\tag{4.12}$$

La Ecuación 4.12 completa este análisis, siendo la conclusión general que el descriptor `mcBRIEF` es invariante a todos los cambios en la iluminación de la escena que contempla el modelo offset-diagonal.

### 4.3.3. Cambios de escala y rotación

El descriptor `mcBRIEF` no incorpora ningún mecanismo especial para obtener invariancia a rotaciones ni a cambios de escala. Sin embargo, dado que muchos de los detectores [1, 2, 21] proveen para cada punto de interés la escala a la que fue detectado, y su orientación, siempre es posible aplicar una transformación a las coordenadas de cada test, o en su defecto rotar y escalar el patch de imagen acordemente, para luego extraer el descriptor con la orientación y escala correctas.

Por otro lado, si bien `mcBRIEF` no es invariante a rotaciones, es bastante robusto a pequeñas variaciones de las coordenadas en las que se calcula cada test, a razón de que los mismos se calculan usando un promedio los valores. Por esta característica, el desempeño del descriptor no se verá demasiado afectado ante pequeñas variaciones de orientación, como veremos más adelante en los resultados experimentales.

Además, como se observa en [2, 3], para obtener invariancia a rotaciones no sólo se debe sacrificar velocidad de ejecución, sino que también puede empeorar el error de reconocimiento en datasets

para los cuales este invariante es prescindible. Por esta razón, a menos que realmente se necesite, la corrección de orientación debería ser omitida. Tal es el caso de aplicaciones donde se espera que todas las imágenes sean capturadas aproximadamente con la misma orientación, como sucedería si la cámara estuviera fijada sobre un robot que se mueve paralelo al suelo.

## Capítulo 5

# Resultados experimentales

Evaluaremos el descriptor `mcBrief` en forma experimental a fin de determinar como influye la modificación propuesta en los resultados prácticos. Dicha evaluación se dividirá en dos partes, siendo la primera en forma independiente del detector, y luego será evaluado su desempeño una aplicación de reconocimiento de imágenes en combinación con un detector estándar.

### 5.1. Configuración de prueba

En todas las pruebas realizadas, a menos que se especifique distinto, se genera un descriptor de 32 bytes de longitud (256 bits), al que nos referiremos como `mcBrief-32`, o `BRIEF-32` según corresponda. El cálculo de cada descriptor se hace a partir de un patch cuadrado de imagen de  $48 \times 48$  píxeles, centrado en el punto de interés. El conjunto de ubicaciones  $(x, y)$  que define el grupo de tests binarios de BRIEF, y adicionalmente el conjunto de vectores  $\alpha$  requerido por `mcBrief`, se encuentra precalculado y se mantiene fijo a lo largo de todas las pruebas (Veáse el Apéndice A para más detalles). El conjunto de puntos  $(x, y)$  que usaremos es el incluido en la implementación de BRIEF disponible en OpenCV [22]. La comparación entre descriptores `mcBrief` y BRIEF se hace mediante la distancia de Hamming.

La extracción descriptores BRIEF a partir de las imágenes color requiere que las mismas se conviertan previamente a escala de grises, lo cual se hace automáticamente en la implementación en uso.

### 5.2. Evaluación del descriptor

Las características del descriptor serán evaluadas, en forma independiente de todo algoritmo detector, por medio de un dataset de imágenes conocido y frecuentemente utilizado en el área, de modo que los resultados obtenidos pueden ser comparados con valores de referencia. El dataset seleccionado tiene la particularidad de que, además de imágenes, contiene una serie de homografías que nos permiten conocer las correspondencias entre puntos en forma precisa, por medio de un cálculo matemático, permitiéndonos prescindir del algoritmo detector y sirviendo de *ground truth* para las pruebas a realizar.

#### 5.2.1. Descripción del dataset

El dataset que usaremos se divide en cinco grupos de imágenes a color (Wall, Trees, Graf, Bikes, Bark, UBC y Leuven), y un grupo de imágenes en escala de grises (Boat) que no consideraremos en este trabajo; la Figura 5.1 muestra una imagen de cada uno de ellos. El dataset completo se encuentra públicamente disponible en la web<sup>1</sup>. Cada grupo se compone de 6 imágenes y 5 homografías que

<sup>1</sup><http://www.robots.ox.ac.uk/~vgg/research/affine/>

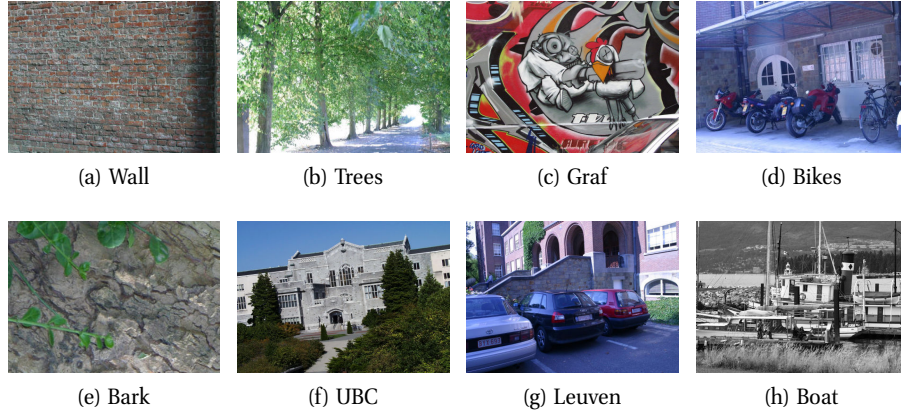


Figura 5.1: Imágenes de referencia en el dataset de evaluación

mapean las coordenadas de una de las imágenes, considerada como imagen de referencia, a cada una de las imágenes restantes en el grupo. Dentro de cada grupo, las imágenes están numeradas en orden de dificultad, de manera que encontrar correspondencias entre las imágenes 1 y 2 es menos complejo que hacerlo entre las imágenes 1 y 6, véase la Figura 5.2 como ejemplo. Entre imágenes de grupos distintos no existe ninguna correspondencia.

En el dataset, cada uno de los grupos está especialmente seleccionado para evaluar el desempeño del descriptor ante un tipo de variación en particular:

- **Ruido:** Trees y Bikes
- **Punto de vista:** Wall y Graf
- **Rotación y escala:** Bark y Boat
- **Cambio en la iluminación de la escena:** Leuven
- **Compresión JPEG:** UBC

### 5.2.2. Distribución de distancias

En esta sección estudiaremos la distribución de distancias para el descriptor `mcBrief`. Esta característica es relevante porque la tarea de determinar si dos descriptores dados corresponden a un mismo punto, se convierte en un problema de clasificación binario, donde cada par de descriptores deberá ser etiquetado como *match* o *nomatch*. Por esta razón, la clasificación será más precisa cuanto mejor separadas se encuentren las distribuciones de las distancias del conjunto *matches* y del conjunto *nomatches*.

El objetivo de la prueba es estimar las distribuciones de probabilidad de cada uno de los grupos usando un histograma normalizado de frecuencias de las distancias entre descriptores. Para tal fin, se generará un número suficientemente grande de descriptores y luego, tomándolos de a pares, se calculará la distancia entre ellos. Similar a como se hace en [3], se construye una grilla rectangular cubriendo la *imagen de referencia* del grupo Wall, Imagen 1, asignando una separación de 12 píxeles entre punto y punto, con lo que se obtienen aproximadamente 4000 puntos distintos. Dado que la homografía entre la imagen de referencia y las demás es conocida, la misma se aplica para mapear la grilla completa a cada una de las 5 imágenes restantes. Posteriormente, se extrae el descriptor de todos los puntos en la grilla de cada imagen, exceptuando a aquellos que se encuentren muy próximos al borde los cuales son descartados.

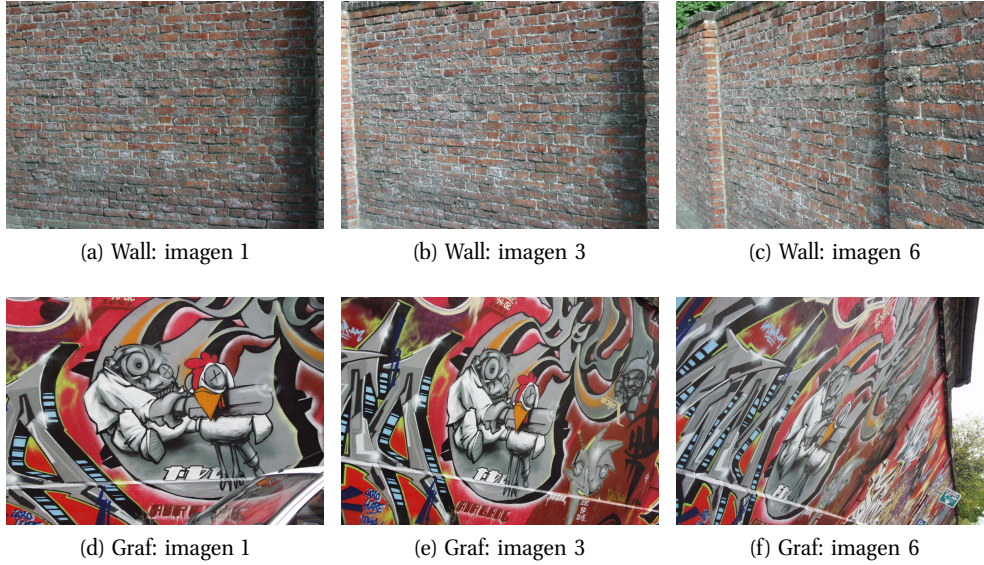


Figura 5.2: Muestra de la creciente dificultad de reconocimiento en los grupos Wall y Graf

Una vez finalizada la etapa de extracción, se toma cada descriptor proveniente de la imagen de referencia y se calcula su distancia a cada uno de los descriptores de las otras imágenes. Debido a que para cada punto en la imagen de referencia se conoce exactamente, por medio de la homografía, cual es su punto correspondiente en cada una de las demás, es posible clasificar todos los pares de descriptores como *match* o *nomatch* sin posibilidad de error, y asignar su distancia al histograma correspondiente.

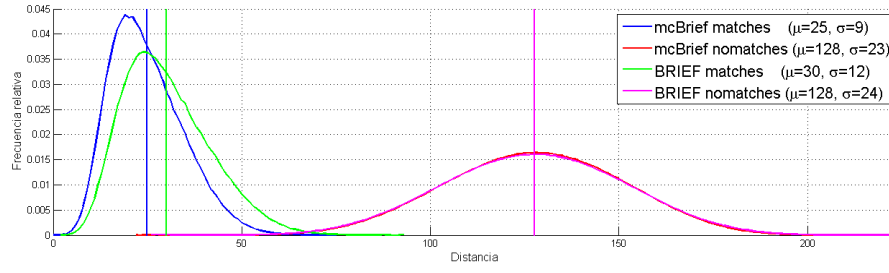
El proceso así descrito se realiza empleando descriptores *mcBrief-32* y descriptores *BRIEF-32*, tomándose estos últimos como medida de comparación. Al ser ambos descriptores de una longitud de 256 bits, la distancia máxima posible entre dos de ellos es de 256 y corresponde al caso en que todos sus bits difieren.

## Resultados

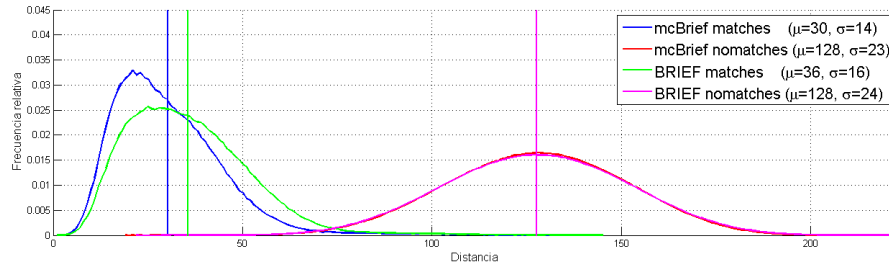
Los histogramas obtenidos con el método detallado anteriormente, para cada uno de los pares de imágenes, pueden verse en la Figura 5.3. Cada gráfico contiene el histograma normalizado del conjunto *matches* y del conjunto *nomatches* para ambos tipos de descriptores, además, las líneas verticales señalan el valor medio de la curva de su mismo color. En la leyenda se detallan el tipo de descriptor y, numéricamente, el valor medio y la desviación estándar de cada una de las curvas.

En todos los casos, se observa claramente que el histograma de *matches* correspondiente al descriptor *mcBrief* se encuentra más cerca en promedio del valor cero y que su desviación estándar es menor o igual a la correspondiente para el descriptor *BRIEF*. A su vez, los histogramas para el conjunto *nomatches* sigue la forma casi exacta de una distribución Gaussiana centrada en el valor 128 que, no casualmente, corresponde exactamente la mitad de la máxima distancia posible. En conclusión, las distribuciones de probabilidad para el descriptor *mcBrief* se encuentran mejor diferenciadas entre sí, posibilitando una mejor clasificación que su contraparte, e indicando que este tipo de descriptor es potencialmente más discriminante que el descriptor *BRIEF*.

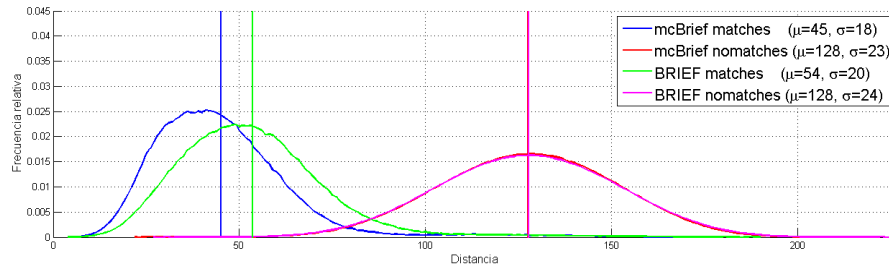
En la Figura 5.3, claramente, se nota también el incremento de dificultad en el *matching* para cada uno de los pares de imágenes. Obsérvese que en la Figura 5.3a los histogramas de los conjuntos de *matches* y *nomatches* se encuentran perfectamente separados, y en los gráficos sucesivos empieza a haber una superposición entre ambos. Alcanzando la superposición máxima para la Figura 5.3e, coincidiendo con el par de imágenes de mayor dificultad en el grupo Wall.



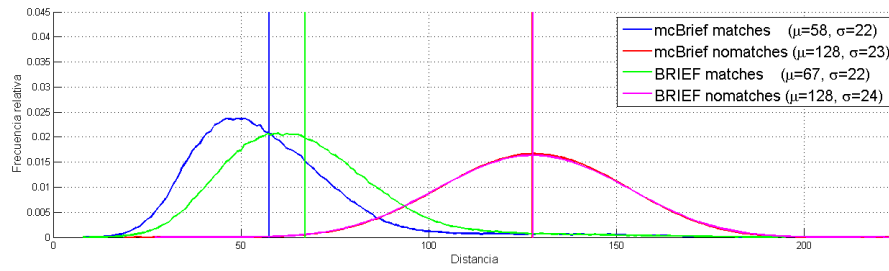
(a) Imagen 1 y 2



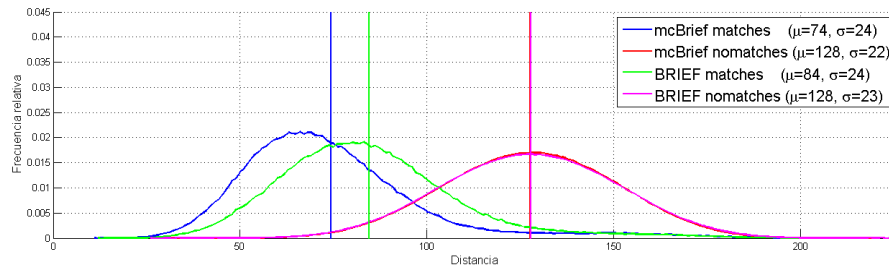
(b) Imagen 1 y 3



(c) Imagen 1 y 4



(d) Imagen 1 y 5



(e) Imagen 1 y 6

Figura 5.3: Histograma normalizado de distancias para el dataset Wall



### 5.2.3. Precision vs. recall

El estudio de las distribuciones de probabilidad de las distancias hecho previamente, permite ver en forma clara el efecto que produjo nuestra modificación sobre el descriptor, pero no permite visualizar adecuadamente el tipo de mejora que se verá realmente a la hora de realizar la tarea de clasificación. Un análisis más apropiado del desempeño del descriptor es mediante las curvas *1-precision vs. recall* como sugirió Mikolajczyk y Schmid en [10].

La configuración de la prueba es similar a la de antes, pero en este caso usaremos todos los datasets de imágenes a color. Hemos agrupados las imágenes según el tipo de variación que contienen para poder evaluar la robustez de nuestro descriptor frente a cada uno de ellos. Los grupos así formados nos permitirán analizar las siguientes características: *resistencia al ruido* (Trees y Bikes), *invariancia a cambios en el punto de vista* (Wall y Graf), *invariancia a rotaciones y cambios de escala* (Bark), *efecto de la compresión JPEG* (UBC), y *efecto de los cambios de iluminación* (Leuven).

## Resultados

Las curvas de *1-precision vs. recall* de cada uno de los grupos se muestran en la Figura 5.4. En todos los grupos, excepto en el de cambios de iluminación, se observa un mejor desempeño del descriptor mcBrief respecto de BRIEF, lo que se visualiza en las curvas como un mayor valor de *recall* para valores de precisión equivalentes. La diferencia mayor se presenta en el grupo de cambios de punto de vista (Figura 5.4b), donde el desempeño de BRIEF es moderado y mcBrief lo supera con creces, un caso similar ocurre para el grupo con compresión JPEG (Figura 5.4d). En el caso del grupo de resistencia al ruido (Figura 5.4a) vemos que descriptor BRIEF obtiene valores altos de *recall* durante la mayor parte de la curva, aún así, el descriptor mcBrief lo supera con un margen considerable.

La curva correspondiente al grupo con cambios de rotación y cambios de escala (Figura 5.4c) presenta valores pobres de *recall*, menor a 0,3 para mcBrief en su mayor parte, pero aún así su mejora es importante respecto de BRIEF, que presenta valores menores a 0,2 en el mismo rango. El hecho de que estos valores sean relativamente bajos en comparación a los demás, no constituye una sorpresa, debido a que ante la ausencia de un algoritmo detector u otro medio que provea para cada punto de interés su valor de escala y orientación, todos los descriptores fueron extraídos en posición vertical y desde patches de imagen de tamaño fijo. En la práctica, cuando alguno de estos factores sea relevante, deberán tomarse los recaudos correspondientes y aplicar las correcciones necesarias al momento de la extracción de descriptores.

Por último, observando la Figura 5.4e, que corresponde al grupo de imágenes con condiciones de iluminación variantes, se tiene que el descriptor BRIEF obtuvo valores de *recall* mayores que nuestro descriptor. Aún así, el desempeño de mcBrief no es para nada malo, con valores de *recall* superiores a 0,97 con una *precisión* de hasta 0,80 y apenas decayendo hasta 0,92 si se desea mejorar la precisión a 0,90. La causa de la desmejoría no la conocemos en este momento, pero será estudiada en el futuro, fuera del alcance de este trabajo.

La conclusión que se desprende es que mcBrief tiene un mejor desempeño general, como se predijo a partir de sus distribuciones de probabilidad, que el descriptor BRIEF en iguales condiciones. También se corroboró la afirmación obvia de que si se esperan grandes variaciones en escala y orientación de las imágenes, debería aplicarse una corrección apropiada a esta situación, y que si el éxito de la tarea a desarrollar es altamente dependiente de la invariancia a cambios de iluminación, deberá considerarse cuidadosamente si conviene el uso de descriptores mcBrief para la misma.

### 5.2.4. Análisis ROC

Similarmente al análisis de *precisión vs. recall*, se puede realizar un análisis de sensibilidad mediante una curva ROC (Receiver Operating Characteristic). A continuación se presenta éste, usando los mismos



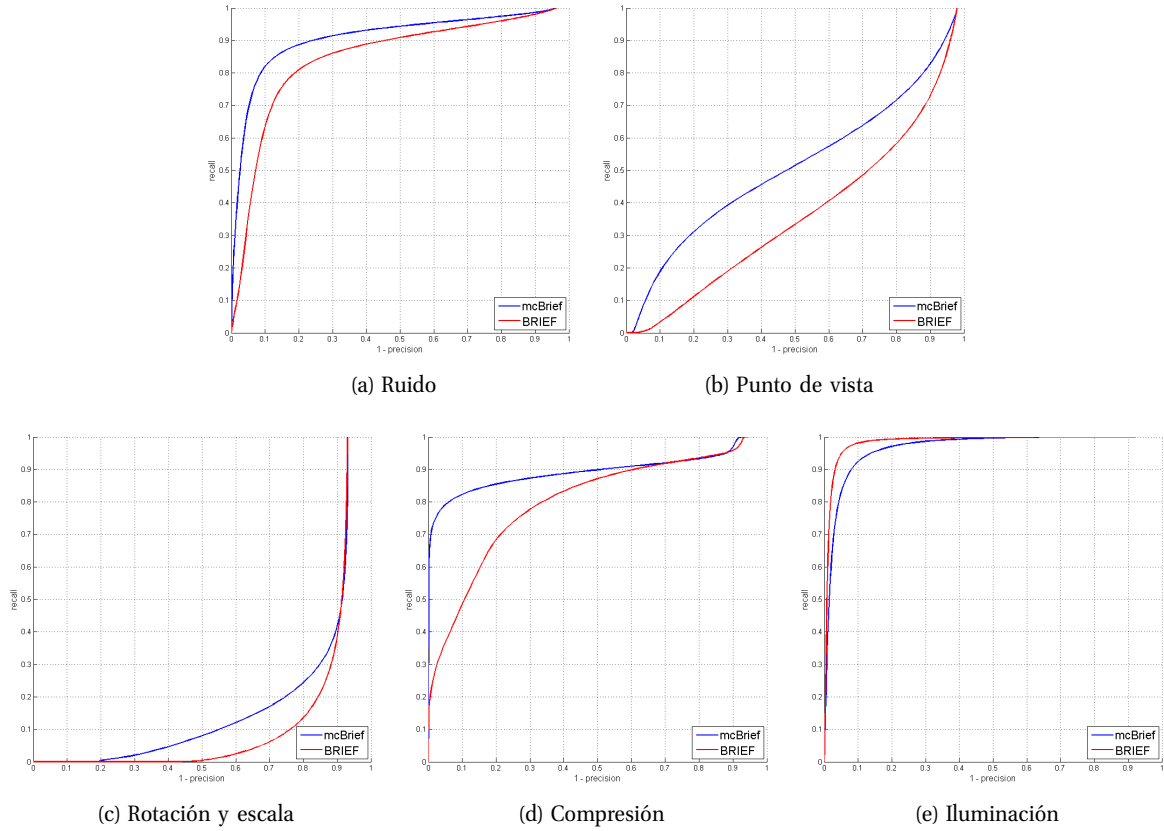


Figura 5.4: gráficos 1-precision vs. recall. (a) Resistencia al ruido. (b) Invariancia a cambios de punto de vista. (c) Invariancia a rotaciones y cambios de escala. (d) Efecto de la compresión JPEG. (e) Cambios de iluminación.

datos que en la experiencia anterior, y complementando sus resultados mediante este otro enfoque. Además, nos interesan particularmente las curvas ROC para poder cuantificar en forma numérica, por medio del cálculo de los valores de AUC (Area Under the Curve) y EER (Equal Error Rate), la diferencia observada gráficamente.

## Resultados

La Figura 5.5 es el gráfico ROC, *False Positive Rate vs. True Positive Rate*, consolidando los datos de todos los datasets. Esta curva representaría el desempeño “promedio” de ambos descriptores. El resultado es muy bueno en los dos casos, pasando ambas curvas próximas al valor ideal de  $(0,1)$ , pero se ve claramente que la correspondiente al descriptor mcBrief siempre está “por encima” de la obtenida para BRIEF, significando un mejor desempeño para el primero.

Los valores de AUC y EER nos sirven para cuantificar por medio de un único escalar el desempeño de los descriptores, estos escalares resumen lo que se visualizaría en las curvas ROC. La diferencia entre ambos es que el valor de AUC resume toda la curva, mientras que el valor de EER representa el punto óptimo de la misma. La Tabla 5.1 contiene los valores de ambas métricas en forma individual para cada uno de los datasets, y el valor consolidado de todos ellos juntos, para cada tipo de descriptor. Cuantificado en forma numérica, el descriptor mcBrief es casi siempre superior que su contraparte, con valores de AUC mayores y menor EER que el descriptor BRIEF, la excepción ocurre para el dataset Leuven como se observó en la sección anterior.

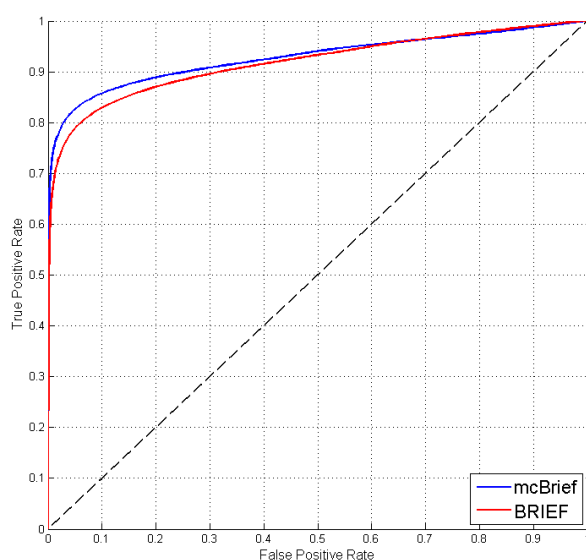


Figura 5.5: Curva ROC consolidada

	AUC		EER %	
	mcBrief	BRIEF	mcBrief	BRIEF
Wall	0.981	0.972	5.15	7.56
Trees	0.974	0.964	6.80	8.57
Graf	0.904	0.861	16.50	21.39
Bikes	0.999	0.998	1.16	1.58
Bark	0.574	0.574	44.55	44.67
UBC	0.968	0.959	8.34	8.69
Leuven	0.998	0.999	1.65	0.88
Todos	0.928	0.918	12.72	14.36

Tabla 5.1: Valores de AUC y ERR

### 5.3. Reconocimiento de imágenes

En una aplicación de reconocimientos imágenes habrá un grupo inicial de imágenes “modelos” con las cuales se construye una base de datos que servirá luego para realizar búsquedas. Más tarde, el usuario proveerá otras imágenes independientes a las usadas como modelo, donde cada una de ellas en forma individual constituirá la “consulta”. Entonces, la aplicación deberá recuperar de la base de datos la imagen modelo que mejor corresponda con la consulta y devolverla como respuesta a la misma. A veces, se prefiere que la respuesta sea un grupo de imágenes modelo, ordenadas según el grado de correspondencia con la consulta. En cualquiera de los casos, el proceso tiene varias diferencias con la configuración de la experiencia planteada en la sección anterior, siendo las más obvias, el que no se posee ningún tipo de información extra para relacionar las coordenadas de la imagen consulta con las coordenadas de la imagen modelo, y el hecho de que el resultado de la búsqueda de correspondencias en la base de datos no será independiente de la cantidad y tipo de imágenes que conformen a la misma. También, el tiempo que se demore en devolver una respuesta puede convertirse en una variable determinante. Diferencias como las expuestas, hacen importante la evaluación del nuevo descriptor dentro de un contexto como el descrito, además de como ya se hizo en forma individual.

Estudiaremos el desempeño del descriptor `mcBrief` emulando lo mejor posible una aplicación real de reconocimiento de imágenes. En esta tarea se emplearán dos datasets de imágenes distintos, el



Figura 5.6: ejemplo de imágenes modelo y consulta en el dataset Caltech-Covers

primero denominado *Caltech-Covers* contiene fotos de CD's desde múltiples puntos de vista y orientación, el segundo dataset se llama *Pasadena-Houses* y está formado por fotos de la fachada de un grupo de edificaciones, tomadas desde diferentes ubicaciones y en distintos momentos del día. Las imágenes en el dataset Caltech-Covers corresponden a la categoría de objetos planares e imágenes de interior, mientras que el dataset Pasadena-Houses contiene fotos de escenas 3D e imágenes de exterior. Usando ambos datasets se abarcan muchos casos de uso típico en problemas reales.

Los resultados obtenidos para el descriptor *mcBrief* se compararán a los que se observan en las mismas condiciones para el descriptor *BRIEF*, para poder evaluar si existe mejora alguna respecto de éste, así como también se compararán con los obtenidos para los descriptores *U-SURF* y *SURF* que representan el estado del arte de los descriptores de features locales. Usaremos el algoritmo 'Fast-Hessian' como detector, en la implementación definida para su uso con el descriptor *SURF*. Los algoritmos de extracción de descriptores y detector corresponden a los que se encuentran en la última versión de *OpenCV* [22].

### 5.3.1. Dataset Caltech-Covers

El dataset Caltech-Covers [23] se divide en dos grupos de imágenes, el primero conteniendo diferentes carátulas de CD's retratadas de frente y ocupando todo el área de la imagen (las mismas fueron descargadas del sitio <http://freecovers.net/>), y el segundo grupo conteniendo cuatro fotografías de cada uno de los CD's desde diferentes ubicaciones y ocupando la imagen sólo parcialmente, este grupo fue usado originalmente en [24].

Las carátulas de CD del primer grupo fueron llevadas a un tamaño de  $600 \times 600$  píxeles, dado que su tamaño varía bastante en la configuración original, y 80 de ellas fueron usadas como imágenes modelo para construir la base de datos de la aplicación (conjunto de entrenamiento). Las fotografías en el segundo grupo poseen todas un tamaño  $640 \times 480$  píxeles, tomadas con una cámara *Canon PowerShot S230*, y no se les realizó ningún tipo de procesamiento adicional. Se eligieron de este grupo las 4 fotografías correspondientes a cada uno de los 80 modelos seleccionados anteriormente, y las mismas se usaron como imágenes de consulta (conjunto de test). La Figura 5.6 muestra ejemplos de ambos grupos de imágenes. La particularidad de este dataset es que el objeto retratado es planar, se encuentra fotografiado sobre diferentes tipos de fondo, y se observa una marcada variación en la escala y orientación en las imágenes de test respecto de los modelos.



Figura 5.7: ejemplo de imágenes modelo y consultas en el dataset Pasadena-Houses

### 5.3.2. Dataset Pasadena-Houses

El dataset Pasadena-Houses [23] consiste en 750 fotografías de fachadas de 102 casas y 23 edificios de la ciudad de Pasadena. Cada tipo de construcción fue fotografiada 6 veces, 3 veces en la tarde y 3 veces por la mañana del día siguiente, para obtener condiciones de iluminación variantes. En todos los casos, las casas y edificios se fotografiaron de frente y desde los laterales, aproximadamente  $30^\circ$  desde la izquierda y  $30^\circ$  desde la derecha. Las fotografías fueron tomadas con dos cámaras de marcas y resoluciones distintas para agregar diversidad.

A los fines de este experimento, hemos elegido las 125 imágenes que fueron tomadas de frente y por la tarde de cada edificación como imágenes modelo, conjunto de entrenamiento, y las 725 imágenes restantes se asignaron al conjunto de test. Todas las imágenes tomadas con una de las cámaras, la de mayor resolución, fueron reducidas hasta un tamaño de  $640 \times 480$  píxeles para que coincidan con las fotografías tomadas con la cámara de menor resolución.

La Figura 5.7 muestra ejemplos del conjunto de entrenamiento y del conjunto de test. El desafío del dataset Pasadena-Houses consiste en que representa diferentes vistas de una escena en 3 dimensiones, las edificaciones no son objetos planares y sus límites no se encuentran definidos claramente como sí sucede en el caso de objetos pequeños. Además, la diversidad de cámaras y las diferencias en iluminación agregan mayor generalidad a la experiencia.

### 5.3.3. Algoritmo de entrenamiento

El algoritmo de entrenamiento es muy simple y consiste únicamente en leer una a una las imágenes seleccionadas como modelo, aplicarles el detector Fast-Hessian [2] para obtener un conjunto de puntos de interés, y extraer un descriptor a partir del mismo (véase el Algoritmo 1). No todos los puntos

devueltos por el detector pueden ser usados para extraer descriptores porque cada descriptor requiere un patch de imagen de donde ser extraído y, para los puntos de interés que se ubican muy próximos al borde de la imagen, este patch requerido puede no estar completo. Por esta razón, se agrega un paso intermedio que consiste en filtrar los puntos devueltos por el detector para eliminar aquellos para los que no sea posible extraer un descriptor. Este paso es dependiente del tipo de descriptor en uso. Una vez extraídos todos los descriptores, estos son almacenados secuencialmente en un archivo, con una referencia indicando el modelo al que pertenecen, formando la base de datos que se usará para búsquedas. Los descriptores que usaremos son mcBrief-32, BRIEF-32, U-SURF-64 y SURF-64.

---

**Algoritmo 1** Entrenamiento

---

**Require:** *ImageTrainSet*

*Database*  $\leftarrow \emptyset$

**for each**  $i \in \text{ImageTrainSet}$  **do**

$\text{keypoints}_i \leftarrow \text{detect}(i)$

$\text{keypoints}'_i \leftarrow \text{filter}(\text{keypoints}_i)$

$\text{descriptors}_i \leftarrow \text{extract}(\text{keypoints}'_i)$

$\text{Database} \leftarrow \text{Database} \cup (i, \text{descriptors}_i)$

**end for**

**return** *Database*

---

#### 5.3.4. Estrategia de matching

El algoritmo de matching tiene el objetivo de buscar en la base de datos el modelo o los modelos que mejor corresponden con la imagen de consulta. En nuestro caso, decidimos simplificar al máximo este paso para poder evaluar el desempeño de los distintos descriptores en condiciones iguales y sin agregar complejidad extra que pueda interferir con los resultados. Por esta razón, hemos elegido la estrategia de búsqueda por fuerza bruta en la base de datos completa, comparando cada descriptor de la imagen de entrenamiento con todos los descriptores existentes en la base de datos. La comparación se realiza por medio de la distancia de Hamming para los descriptores mcBrief y BRIEF, y por medio de la distancia Euclidiana para los descriptores U-SURF y SURF. En este proceso se utiliza el *nearest neighbor distance ratio* [10, 2] con un *threshold* de 0,7. Es decir, un descriptor de la imagen de entrenamiento  $d_T$  y su vecino más próximo en la base de datos  $d_1$  sólo se considera un match si, siendo  $d_2$  el segundo vecino más próximo, se cumple:

$$\text{match}(d_T, d_1) = \text{True} \Leftrightarrow \left[ \text{model}(d_1) \equiv \text{model}(d_2) \vee \frac{\text{dist}(d_T, d_1)}{\text{dist}(d_T, d_2)} < 0,7 \right]. \quad (5.1)$$

Cada par de descriptores que forma un match contribuye un voto para el modelo asociado a  $d_1$ . Finalmente, usando votación simple se ordenan los modelos de la base de datos desde el más probable al menos probable. En el Algoritmo 2 puede verse el pseudocódigo del algoritmo de matching.

#### 5.3.5. Resultados

Analizaremos el resultado de la prueba de reconocimiento planteada desde dos aspectos, primero considerando el éxito para identificar la imagen modelo correcta en cada consulta, y segundo con respecto al tiempo que se demora en obtener una respuesta.



**Algoritmo 2** Matching

---

**Require:** *ImageTestSet*, *Database*  
 $Vote \leftarrow \emptyset$   
**for each**  $m \in models(Database)$  **do**  
     $Vote[m] \leftarrow 0$   
**end for**  
**for each**  $i \in ImageTestSet$  **do**  
     $keypoints_i \leftarrow detect(i)$   
     $keypoints'_i \leftarrow filter(keypoints_i)$   
     $descriptors_i \leftarrow extract(keypoints'_i)$   
    **for each**  $d_T \in descriptors_i$  **do**  
         $d_1 \leftarrow getFirstNeighbor(d_T, Database)$   
         $d_2 \leftarrow getSecondNeighbor(d_T, Database)$   
        **if**  $dist(d_T, d_1) < 0,7 \times dist(d_T, d_2)$  **then**  
             $Vote[model(d_1)] \leftarrow Vote[model(d_1)] + 1$   
        **end if**  
    **end for**  
**end for**  
**return**  $sort(Vote)$

---

**Éxito de reconocimiento**

El éxito de reconocimiento se medirá como el porcentaje de matches correctos respecto del total de imágenes consultadas:

$$recognition\_rate = \frac{\# \text{ matches correctos}}{\text{total imágenes consultadas}} \times 100. \quad (5.2)$$

Teniendo en cuenta que la salida del algoritmo de matching es un ranking de modelos ordenados según su relevancia con la consulta, consideraremos tres tipos de casos como matches correctos:

1. **Primer match:** se considerarán como matches correctos sólo los casos en que el modelo correspondiente sea el primero en el ranking devuelto por la función de matching.
2. **Primer y segundo match:** se considerarán como matches correctos los casos en que el modelo correspondiente sea el primero o el segundo en el ranking.
3. **Del primer al quinto match:** se considerarán como matches correctos los casos en que el modelo correspondiente se encuentre entre los cinco primeros modelos en el ranking.

El resultado para todos los descriptores bajo estudio puede verse en la Tabla 5.2, para el dataset Caltech-Covers, y en la Tabla 5.3, para el dataset Pasadena-Houses. En el primer caso, el mejor desempeño lo obtuvo el descriptor SURF, siguiéndole mcBrief, U-SURF y BRIEF. Es razonable que SURF haya obtenido el primer lugar debido a que es el único, de los cuatro tipos de descriptores, que incorpora corrección de orientación, y las imágenes de test en el dataset Caltech-Covers varían bastante en orientación respecto de los modelos en la base de datos. Sin embargo, nuestro descriptor mcBrief obtuvo el segundo lugar, siendo el mejor de los tres descriptores sin corrección de orientación, y superando por más de un 10 % en todos los casos al descriptor BRIEF del cual deriva.

En el dataset Pasadena-Houses, donde todas las imágenes han sido tomadas aproximadamente en forma horizontal, el mejor resultado lo obtuvo mcBrief, con un éxito de reconocimiento mayor al 91 % en el primer match, siguiéndole U-SURF, BRIEF, y SURF. La diferencia entre el éxito de mcBrief

y BRIEF sigue siendo algo más del 10 %, como en el caso anterior. Un corolario de estos resultados es que, cuando se conoce de antemano que la rotación no es factor determinante, no debería aplicarse corrección de orientación a los descriptores porque podría degradar su desempeño, como ocurrió aquí en el caso de SURF y U-SURF, y como fue anticipado en [2, 3].

	Base de Datos		Test	Match correcto		
	Imágenes	Keypoints		lero.	lero. o 2do.	lero. al 5to.
mcBrief	80	112974	320	260 (81.25 %)	266 (83.13 %)	282 (88.13 %)
BRIEF	80	123494	320	226 (70.63 %)	230 (71.88 %)	249 (77.81 %)
U-SURF	80	191438	320	259 (80.94 %)	265 (82.83 %)	276 (86.25 %)
SURF	80	191438	320	317 (99.06 %)	318 (99.38 %)	320 (100.00 %)

Tabla 5.2: Resultados para el dataset Caltech-Covers

	Base de Datos		Test	Match correcto		
	Imágenes	Keypoints		lero.	lero. o 2do.	lero. al 5to.
mcBrief	125	132995	625	574 (91.84 %)	591 (94.56 %)	606 (96.96 %)
BRIEF	125	148067	625	524 (83.84 %)	554 (92.48 %)	578 (92.48 %)
U-SURF	125	240402	625	562 (89.92 %)	585 (93.60 %)	602 (96.32 %)
SURF	125	240402	625	524 (83.84 %)	550 (88.00 %)	570 (91.20 %)

Tabla 5.3: Resultados para el dataset Pasadena-Houses

### Velocidad de respuesta

En una aplicación de reconocimiento de imágenes es importante el tiempo que se demore en devolver una respuesta, principalmente si se quieren procesar datos en tiempo real. Una forma de cuantificar cuan rápido responderá el sistema es por medio del *tiempo medio de extracción* y *tiempo medio de comparación*:

$$\text{tiempo medio de extracción} = \frac{\# \text{ descriptores extraídos}}{\text{tiempo total de extracción}}, \quad (5.3)$$

$$\text{tiempo medio de comparación} = \frac{\# \text{ comparaciones realizadas}}{\text{tiempo total de comparación}}. \quad (5.4)$$

A partir de los datos recolectados para ambos datasets hemos calculados el valor de ambas mediciones, las que se muestran en la Tabla 5.4. Como parámetro de referencia, tómese en cuenta que los datos han sido recolectados en una notebook *DELL XPS 1340* (Procesador Intel Core2 Duo P8600 y 4GB de memoria RAM).

	Tiempo promedio	
	Extracción [ $\mu$ -secs]	Comparación [ $n$ -secs]
mcBrief-32	56	74
BRIEF-32	25	74
U-SURF-64	169	330
SURF-64	494	330

Tabla 5.4: Tiempo promedio de extracción y comparación de descriptores

Los resultados coinciden con los esperados, el tiempo de comparación de descriptores `mcBrief` y `BRIEF` coincide, debido a que ambos son de la misma longitud y se comparan mediante la distancia de Hamming, y es significativamente menor al tiempo de comparación de los descriptores `U-SURF` y `SURF`, los cuales a la vez coinciden entre sí. Además, el tiempo medio de extracción del descriptor `SURF` es casi diez veces superior al de `mcBrief`, el cual a su vez es el doble del tiempo de extracción del descriptor `BRIEF`. Nótese que, si el procesador usado contara con el juego de instrucciones `SSE4.2` [16] el tiempo de cálculo de la distancia de Hamming se reduciría aún más, y que la diferencia en tiempo de extracción de `mcBrief` con respecto a `BRIEF` se debe principalmente al mayor número de veces que `mcBrief` debe acceder a la imagen integral para cada uno de los tests binarios.



## Capítulo 6

# Conclusión y trabajos futuros

La inquietud que motivó este trabajo fue la observación de que habitualmente capturamos y almacenamos imágenes conteniendo millones de colores pero, paradójicamente, muchos de los algoritmos de *Computer Vision* tienen como primer requisito transformar estas imágenes a un formato de escala de grises, descartando toda información de color. A partir de esta observación, nos propusimos modificar algún algoritmo conocido para permitirle aprovechar la información de color presente en las imágenes y anteriormente ignorada. Así surgió el descriptor `mcBrief`, como una extensión del descriptor BRIEF, que al aplicarse a imágenes a color tiene un desempeño mejor que su predecesor. Este solo hecho, que se demostró en forma práctica a partir de los resultados del Capítulo 5, basta para cumplir el objetivo principal de la tesina.

Analizando los resultados más en detalle, existen varios otros puntos a mencionar. Recordando que las principales ventajas que posee BRIEF respecto de SURF, descriptor estándar *de facto*, es su diferencia en velocidad de extracción y matching, posibilitando el uso de BRIEF en aplicaciones de tiempo real, y su tamaño ocho veces menor que el de un descriptor SURF; fue de nuestro interés tratar de conservarlas. Así es que el descriptor `mcBrief` propuesto mantiene la misma longitud, conservando la velocidad de matching de BRIEF, a diferencia de enfoques comunes en la bibliografía, donde la extensión natural para aplicar los descriptores en imágenes a color consiste en extraer un descriptor de cada canal de color, triplicando el tamaño del descriptor resultante. La extracción de `mcBrief` es algo más compleja, porque se debe acceder a un mayor número de imágenes integrales en cada test, pero esto se compensa porque se omite la conversión de la imagen completa a un formato de un solo canal requerida por BRIEF, de modo que el tiempo promedio de extracción apenas si llega a duplicarse (Sección 5.3.5), manteniéndose aún mucho menor que el tiempo de extracción de descriptores SURF. Esto nos lleva a decir que, salvo en casos con requerimientos de tiempo muy estrictos, el descriptor `mcBrief` puede reemplazar siempre al descriptor BRIEF sin pérdida de eficiencia y mejorando su desempeño en aproximadamente un 10 %, medido por su éxito de reconocimiento (Sección 5.3.5).

Las pruebas en reconocimiento del Capítulo 5, además de los resultados ya comentados, probaron que en datasets donde no se esperan variaciones considerable de orientación, como en el caso de fotografías de paisajes urbanos, el descriptor `mcBrief` es más óptimo que SURF, y considerablemente más rápido que éste. El descriptor `mcBrief` también se convierte en un reemplazo obvio para aplicaciones que estuvieran funcionando con U-SURF, siendo que este último no posee ninguna ventaja adicional, es considerablemente más lento, y tiene un peor desempeño que `mcBrief`.

Algunas cuestiones que deberán ser estudiadas en trabajos futuros son, la leve pérdida de desempeño observada para el dataset con condiciones de iluminación variante (Sección 5.2.3), y la mejor forma lidiar con los cambios de orientación para obtener un descriptor invariante a la rotación de las imágenes. Cabe destacar que este último aspecto también fue observado y dejado pendiente por los autores de BRIEF en [3]. Otra alternativa que nos interesa explorar en el futuro es si, una extensión similar a la usada en `mcBrief` puede también aplicarse y mejorar el desempeño de otros tipos de

descriptores que deban ser extraídos desde imágenes a color. Por último, también dejamos por analizar, si la mejora observada para `mcBrief` en imágenes a color puede optimizarse aún más, extrayendo descriptores `mcBrief` desde otros tipos de imágenes multicanal, más allá del formato RGB ya evaluado.

En resumen, nos hemos propuesto extender un descriptor existente de modo de aprovechar mejor la información presente en imágenes a color, lo que dio origen al descriptor `mcBrief`. Nos pusimos como objetivo conservar las ventajas de eficiencia y economía de representación de BRIEF, lo cual consideramos cumplido. Hemos verificado en forma práctica que el nuevo descriptor obtiene un significativamente mayor éxito de reconocimiento que BRIEF, que U-SURF y, hasta en algunos casos de aplicación, mejor que el descriptor representante del estado del arte, SURF. Finalmente, hemos aislado e identificado perfectamente los puntos débiles de nuestro descriptor, para que puedan ser estudiados en el futuro, y tenidos en cuenta a la hora de diseñar aplicaciones.

## Apéndice A

### Definición de tests binarios

La Tablas A.1, A.2, A.3, A.4, A.5 y A.6 contienen el listado completo de coordenadas  $\{x_1, y_1, x_2, y_2\}$  y coeficientes  $\{\alpha_1, \alpha_2, \alpha_3\}$  que definen los 256 tests de cada descriptor mcBrief-32, tal cual como se usó para obtener los resultados presentados en este trabajo.

#	$x_1$	$y_1$	$x_2$	$y_2$	$\alpha_1$	$\alpha_2$	$\alpha_3$
1	-1	-2	-1	7	0,379836	0,491505	0,693393
2	-1	-14	3	-3	1,344934	-0,647571	-0,238778
3	-2	1	2	11	0,572809	0,992771	0,964736
4	6	1	-7	-10	-1,410099	-0,790453	-0,963915
5	2	13	0	-1	-1,618990	1,227239	0,054158
6	5	-14	-3	5	0,184551	-2,201388	0,877596
7	8	-2	4	2	-0,629316	1,487511	-0,361007
8	8	-11	5	-15	-0,943766	-0,703724	-0,247715
9	-23	-6	-9	8	-2,246945	-0,184619	-0,880271
10	6	-12	8	-10	-0,736674	-0,143768	0,287393
11	-1	-3	1	8	0,852354	0,311852	-0,292251
12	6	3	6	5	-0,654783	0,291563	-0,193424
13	-6	-7	-5	5	-0,005505	1,617503	1,715310
14	-2	22	-8	-11	-0,638405	-0,082244	0,503353
15	7	14	5	8	0,225934	0,252172	-0,564796
16	14	-1	-14	-5	-0,136507	0,561620	0,475404
17	9	-14	0	2	1,227318	0,261245	-1,932577
18	-3	7	6	22	-1,080757	-0,244457	-1,811694
19	6	-6	-5	-8	0,364652	0,600111	-0,621481
20	9	-5	-1	7	0,151201	-0,924359	-0,998403
21	-7	-3	-18	-10	-1,749840	-0,385429	-0,452353
22	-5	4	11	0	0,314559	0,506632	0,847963
23	3	2	10	9	-1,053134	0,030291	0,331427
24	3	-10	9	4	-1,343928	0,642185	-1,835073
25	12	0	19	-3	-0,080484	-0,507129	-0,333855
26	15	1	-5	-11	1,175645	-0,822027	1,855597
27	-1	14	8	7	0,499407	0,033044	-0,761770
28	-23	7	5	-5	1,505585	-0,206171	-2,790185
29	-6	0	17	-10	-0,416338	-1,448265	-2,191323

Tabla A.1: Tests binarios mcBrief-32 (#1-#29)

#	$x_1$	$y_1$	$x_2$	$y_2$	$\alpha_1$	$\alpha_2$	$\alpha_3$
30	-4	13	-4	-3	0,006836	-0,028193	-1,238725
31	1	-12	2	-12	0,510803	-0,078571	0,693116
32	8	0	22	3	0,881928	-1,545890	-0,628931
33	13	-13	-1	3	-0,680775	-2,083796	0,231868
34	17	-16	10	6	-0,414155	-0,525883	-0,192139
35	15	7	0	-5	-0,102137	1,630377	-0,469374
36	-12	2	-2	19	-0,065598	0,422367	1,471046
37	-6	3	-15	-4	0,310069	2,000552	-1,465338
38	3	8	14	0	0,401303	-0,985174	-1,363862
39	-11	4	5	5	-0,373781	0,079900	-0,315795
40	-7	11	1	7	0,212754	-1,122866	0,921608
41	12	6	3	21	0,873321	1,703615	1,139572
42	2	-3	1	14	-1,199357	-1,218463	-2,025012
43	1	5	11	-5	-0,520823	0,177847	0,559963
44	-17	3	2	-6	-2,371698	0,486343	-0,822684
45	8	6	-10	5	1,274322	-1,054334	-0,300204
46	-2	-14	4	0	1,004019	1,161921	-1,962259
47	-7	5	5	-6	-0,493552	-0,662996	-0,397838
48	4	10	-7	4	-0,490121	-1,324687	-0,892472
49	0	22	-18	7	2,164541	0,120638	-0,778428
50	-3	-1	18	0	0,199835	-0,223401	1,443653
51	22	-4	3	-5	-1,590829	0,045063	1,153170
52	-7	1	-3	2	1,233529	-0,468343	0,529844
53	-20	19	-2	17	0,426290	1,093754	-1,219677
54	-10	3	24	-8	1,230573	0,076461	-1,242167
55	-14	-5	5	7	1,311131	0,606235	0,290646
56	12	-2	-15	-4	-2,353677	-1,468214	-1,964321
57	12	4	-19	0	-0,253328	1,168050	-0,098231
58	13	20	5	3	2,390921	-0,545031	-0,770149
59	-12	-8	0	5	0,027139	1,225490	0,366613
60	6	-5	-11	-7	-0,556282	-1,083733	-1,211231
61	-11	6	-22	-3	0,076021	0,107920	-0,186730
62	4	15	1	10	1,306825	-2,468751	-1,798427
63	-4	-7	-6	15	-0,083787	-0,628729	0,066485
64	10	5	24	0	1,569772	0,780516	-0,237118
65	6	3	-2	22	-0,803857	0,262102	-1,747300
66	14	-13	-4	4	-1,308110	1,157216	-0,201474
67	8	-13	-22	-18	-0,739019	0,479408	0,097964
68	-1	-1	3	-7	-0,564669	-1,242351	-0,900872
69	-12	-19	3	4	0,207799	0,430176	0,805628
70	10	8	-2	13	1,930024	-0,042124	-0,692586
71	-1	-6	-5	-6	-1,325145	1,800645	-1,888189
72	-21	2	2	-3	-1,838084	0,529771	-0,382573
73	-7	4	16	0	-0,807689	0,985475	-0,981895
74	-5	-6	-1	-12	-1,657282	0,488328	-0,722377
75	-1	1	18	9	0,168887	-0,443087	-1,530954

Tabla A.2: Tests binarios mcBrief-32 (#30-#75)

#	$x_1$	$y_1$	$x_2$	$y_2$	$\alpha_1$	$\alpha_2$	$\alpha_3$
76	10	-7	6	-11	-0,092378	0,717072	0,810932
77	3	4	-7	19	-0,908952	0,792181	-1,383192
78	5	-18	5	-4	0,567118	0,933262	-0,139302
79	0	4	4	-20	-1,428858	-0,232294	-0,297690
80	-11	7	12	18	-0,252913	-1,422274	0,527676
81	17	-20	7	-18	0,265389	0,636815	-0,991672
82	15	2	-11	19	-0,216041	-0,520587	0,395149
83	6	-18	3	-7	0,082304	1,223193	0,899456
84	1	-4	13	-14	-0,437578	-0,496436	-0,245768
85	3	17	-8	2	-0,587859	1,397524	-0,438045
86	2	-7	6	1	-0,275512	1,789790	0,441906
87	-9	17	8	-2	0,015547	-1,000972	-0,115425
88	-6	-8	12	-1	-0,563375	0,565648	1,881118
89	4	-2	6	-1	0,075856	0,989217	0,125529
90	7	-2	8	6	0,859906	0,340110	-2,070953
91	-1	-8	-9	-7	-2,179974	-0,924325	0,687991
92	-9	8	0	15	-0,484605	-1,336339	0,666750
93	22	0	-15	-4	0,085771	1,225647	-1,088587
94	-1	-14	-2	3	-0,072332	0,371191	-0,661095
95	-4	-7	-7	17	1,859888	-1,578985	0,804284
96	-2	-8	-4	9	0,895788	-1,124568	-0,992826
97	-7	5	7	7	0,562864	1,023370	-0,580840
98	13	-5	11	-8	0,732701	-0,620561	0,217852
99	-4	11	8	0	-1,032474	0,970231	-1,172700
100	-11	5	-6	-9	0,658316	0,978605	0,229974
101	-6	2	-20	3	-0,309916	-0,785756	-0,059156
102	2	-6	10	6	-1,961948	1,133106	1,740560
103	-6	-6	7	-15	-2,152943	0,795326	0,431751
104	-3	-6	1	2	0,746149	-1,540221	-0,625980
105	0	11	2	-3	-0,649754	0,324298	0,877179
106	-12	7	5	14	-0,023672	0,814569	-0,500965
107	-7	0	-1	-1	-0,336011	0,942943	1,050924
108	0	-16	8	6	-1,040160	-0,439462	-1,550021
109	11	22	-3	0	-0,536736	0,938195	1,039625
110	0	19	-17	5	0,416152	-1,307110	0,410262
111	-14	-23	-19	-13	-2,243006	-1,006493	0,604075
112	10	-8	-2	-11	0,305987	0,134517	-0,422462
113	6	-11	13	-10	-0,909260	-0,158401	0,737019
114	-7	1	0	14	0,910250	-0,441433	-0,078676
115	1	-12	-5	-5	-0,053674	0,941259	0,314624
116	7	4	-1	8	-0,152543	0,800386	-0,665392
117	-5	-1	2	15	-0,972064	-0,613078	0,749818
118	-1	-3	-10	7	-1,124629	-0,595383	-0,075925
119	-6	3	-18	10	1,129738	1,303457	0,782327
120	-13	-7	10	-13	-1,049643	0,519209	0,725121

Tabla A.3: Tests binarios mcBrief-32 (#76-#120)

#	$x_1$	$y_1$	$x_2$	$y_2$	$\alpha_1$	$\alpha_2$	$\alpha_3$
121	-1	1	-10	13	1,029026	-0,477383	-0,961980
122	14	-19	-14	8	1,051364	-1,298359	-1,285341
123	-13	-4	1	7	0,091613	0,234767	-1,022222
124	-2	1	-7	12	0,506268	-0,524505	-0,937433
125	-5	3	-5	1	-0,395075	-1,101076	-1,091520
126	-2	-2	-10	8	-0,523777	-0,104320	-0,842513
127	14	2	7	8	1,293057	-0,002162	-0,074197
128	9	3	2	8	-2,638776	0,104853	-0,740284
129	1	-9	0	-18	-0,409722	-0,637499	0,959907
130	0	4	12	1	-0,227601	0,106114	0,322249
131	9	0	-10	-14	-0,782633	0,051537	1,134804
132	-9	-13	6	-2	1,379581	-0,089178	0,441009
133	5	1	10	10	-0,047812	0,088917	0,642666
134	-6	-3	-5	-16	0,529390	-0,360826	0,569715
135	6	11	0	-5	-0,463209	1,337367	-0,966907
136	10	-23	2	1	0,313150	0,546431	-0,708523
137	-5	13	9	-3	-2,448463	-0,800587	-0,087454
138	-1	-4	-5	-13	1,239921	1,532458	-0,171929
139	13	10	8	-11	0,726325	-1,848522	0,834786
140	20	19	2	-9	-0,731739	-0,474815	-0,294715
141	-8	4	-9	0	0,948501	0,805423	-1,060595
142	10	-14	19	15	1,009198	1,755495	0,344860
143	-12	-14	-3	-10	-1,256755	-0,597788	0,260969
144	-3	-23	-2	17	0,670984	2,084898	2,486532
145	-11	-3	-14	6	0,296047	1,550930	1,392081
146	-2	19	2	-4	-0,532076	0,940511	0,588584
147	5	-5	-13	3	-0,089483	-0,179558	1,071616
148	-2	2	4	-5	1,858112	0,051225	-0,247085
149	4	17	-11	17	-0,181465	1,808774	-1,752799
150	-2	-7	23	1	-0,709739	1,389037	1,118748
151	13	8	-16	1	0,604824	0,315818	0,128146
152	-5	-13	-17	1	1,041152	0,188313	1,596955
153	6	4	-3	-8	-0,056547	1,558883	1,006787
154	-9	-5	-10	-2	-0,229296	0,659754	0,847715
155	0	-9	-2	-7	1,080447	0,044861	-0,345248
156	0	5	2	5	1,686136	0,185875	0,879400
157	-16	-4	3	6	-0,227654	0,542751	-0,394322
158	-15	2	12	-2	0,723806	-0,587327	0,270836
159	-1	4	2	6	-1,892740	-0,421801	-0,064452
160	1	1	-8	-2	-0,703149	-0,775986	0,324814
161	12	-2	-2	-5	-0,655524	0,350558	2,344950
162	8	-8	9	-9	-1,385859	-0,180004	-0,144040
163	-10	2	1	3	0,040294	0,475785	-0,400891
164	10	-4	4	-9	-0,839364	-1,215140	1,315506
165	12	6	5	2	-1,028715	-0,730324	-0,673502
166	-8	-3	5	0	1,110307	0,968441	-0,035148

Tabla A.4: Tests binarios mcBrief-32 (#121-#166)

#	$x_1$	$y_1$	$x_2$	$y_2$	$\alpha_1$	$\alpha_2$	$\alpha_3$
167	1	-13	2	-7	0,090191	0,303263	-1,233949
168	-10	-1	-18	7	1,141898	1,463209	0,320344
169	8	-1	-10	-9	0,304025	1,059397	0,514929
170	-1	-23	2	6	1,856162	-0,006661	0,569382
171	-3	-5	2	3	0,618597	0,527169	-0,004017
172	11	0	-7	-4	0,651861	-0,074142	0,611754
173	2	15	-3	-10	0,937341	-0,303468	-0,490172
174	-8	-20	3	-13	-0,873093	-0,956165	-1,727182
175	-12	-19	-11	5	-1,050417	0,634454	-0,557286
176	-13	-17	2	-3	1,075945	0,715344	0,802125
177	4	7	0	-12	0,903193	-0,217214	1,503910
178	-1	5	-6	-14	-0,816329	-0,140625	0,919659
179	11	-4	-4	0	-0,106564	0,212877	-0,557247
180	10	3	-3	7	-0,014247	0,825391	0,348651
181	21	13	6	-11	0,581500	-1,266023	-1,088510
182	24	-12	-4	-7	0,342838	-0,112147	0,600643
183	16	4	-14	3	-1,466746	0,550873	0,413025
184	5	-3	-12	-7	0,952964	0,564105	-1,524290
185	-4	0	-5	7	0,096192	0,792628	-0,311119
186	-9	-17	-7	13	0,198629	0,302711	1,636788
187	-6	22	5	-11	0,305555	-0,300551	-2,002543
188	-8	2	-11	23	0,204119	-0,571996	-0,656133
189	-10	7	14	-1	-0,273076	1,333944	-1,572005
190	-10	-3	3	8	-0,998549	0,658449	-1,051080
191	1	-13	0	-6	-0,294659	0,945295	-1,653799
192	-21	-7	-14	6	-0,417273	1,395046	0,369416
193	19	18	-6	-4	0,830244	0,127521	-0,936180
194	7	10	-4	-1	2,404791	-0,310309	0,098282
195	21	-1	-5	1	-0,699038	0,241006	0,630276
196	6	-10	-2	-11	0,843313	0,629462	-0,321484
197	-3	18	7	-1	0,431178	-1,766886	0,627670
198	-9	-3	10	-5	0,154274	0,867203	-0,211632
199	14	-13	-3	17	0,750126	-1,687284	-0,288022
200	-19	11	-18	-1	0,472205	-0,110524	0,342103
201	-2	8	-23	-18	0,115759	-0,228526	0,979721
202	-5	0	-9	-2	-1,289186	-1,947936	-0,342678
203	-11	-4	-8	2	-0,778689	-0,455493	1,647245
204	6	14	-6	-3	1,387012	0,498487	-1,244250
205	0	-3	0	-15	0,804394	0,834723	-2,124676
206	4	-9	-9	-15	0,469655	-0,965685	0,225550
207	11	-1	11	3	1,323262	-1,307886	-2,321065
208	-16	-10	7	-7	0,447652	1,358313	-2,897502
209	-10	-2	-2	-10	0,077205	0,083792	-0,162138
210	-3	-5	-23	5	-0,300574	-0,624079	0,114268
211	-8	13	-11	-15	-0,509345	-0,314951	-0,419009
212	11	-15	-6	6	-0,079462	-0,623169	-0,360430

Tabla A.5: Tests binarios mcBrief-32 (#167-#212)

#	$x_1$	$y_1$	$x_2$	$y_2$	$\alpha_1$	$\alpha_2$	$\alpha_3$
213	-3	-16	2	-2	-1,508997	0,281906	0,454457
214	12	6	24	-16	-0,286185	0,666644	-0,477513
215	0	-10	11	8	1,372396	-1,651512	0,118539
216	7	-7	-7	-19	-0,170626	-1,962366	-0,419189
217	16	5	-3	9	0,176609	-0,044805	-0,484547
218	7	9	-16	-7	0,274156	-0,392481	1,751871
219	2	3	9	-10	2,199409	1,404053	1,881204
220	1	21	7	8	-1,047529	-0,222299	0,180392
221	0	7	17	1	-1,360769	-0,284890	0,796307
222	12	-8	6	9	0,640337	-1,046240	1,254321
223	-7	11	-6	-8	0,820909	1,465587	-1,585317
224	0	19	3	9	-0,543960	1,112901	-2,420266
225	-7	1	-11	-5	-0,917934	-0,332556	1,185455
226	8	0	14	-2	-0,189267	-0,369482	0,044885
227	-2	12	-6	-15	-0,713667	0,733163	-1,441928
228	12	4	-21	0	0,497225	-1,194453	0,690471
229	-4	17	-7	-6	0,201161	-0,448237	0,731628
230	-9	-10	-7	-14	-3,331763	-0,551738	1,891626
231	-10	-15	-14	-15	0,123951	-0,118793	-0,217132
232	-5	-7	-12	5	-0,985511	2,515453	-0,208948
233	0	-4	-4	15	-0,751127	-1,267808	-1,138784
234	2	5	-23	-6	0,222708	-1,508117	-2,585435
235	-21	-4	4	-6	-0,620371	1,635447	-0,709736
236	5	-10	6	-15	-1,338493	-0,054596	1,137866
237	-3	4	5	-1	-0,198232	0,626701	-1,080833
238	19	-4	-4	-23	-0,746183	-1,085807	0,232217
239	17	-4	-11	13	-0,434169	-0,740335	-1,482294
240	12	1	-14	4	-1,680043	0,164976	1,449531
241	-6	-11	10	-20	0,893113	-0,446857	1,278263
242	5	4	20	3	-0,234018	1,550499	-0,861916
243	-20	-8	1	3	0,114707	0,547394	-1,056562
244	9	-19	-3	9	-1,204533	-0,456669	0,966733
245	15	18	-4	11	-0,157438	-0,673552	1,064580
246	16	12	7	8	1,283404	-0,389167	0,913014
247	-8	-14	9	-3	-0,082462	0,708986	0,026659
248	0	-6	-4	2	1,124397	0,600096	0,312381
249	-10	1	2	-1	0,565262	0,803418	0,467924
250	-7	8	18	-6	0,176129	-1,140188	-0,430092
251	12	9	-23	-7	0,017211	-0,375437	1,954803
252	-6	8	2	5	-0,284064	1,133133	-0,933775
253	6	-9	-7	-12	0,979672	-1,485738	-0,738375
254	-2	-1	2	-7	0,542885	0,458659	0,029368
255	9	9	15	7	-1,328249	-1,025168	0,081517
256	2	6	6	-6	1,394660	0,844326	-1,083309

Tabla A.6: Tests binarios mcBrief-32 (#213-#256)



# Bibliografía

- [1] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60:91–110, November 2004.
- [2] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). *Comput. Vis. Image Underst.*, 110:346–359, June 2008.
- [3] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: binary robust independent elementary features. In *Proceedings of the 11th European conference on Computer vision: Part IV, ECCV'10*, pages 778–792, Berlin, Heidelberg, 2010. Springer-Verlag.
- [4] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010.
- [5] J. von Kries. Evaluating color descriptors for object and scene recognition. In D. L. MacAdam, editor, *Sources of Color Science*, pages 109–119, Cambridge MA, 1970. The MIT Press.
- [6] G.D. Finlayson, S.D. Hordley, and R. Xu. Convex programming colour constancy with a diagonal-offset model. In *ICIP05*, pages III: 948–951, 2005.
- [7] R. Szeliski. *Computer Vision: Algorithms and Applications*. Online, September 2010. Book website: <http://szeliski.org/Book/>.
- [8] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:511, 2001.
- [9] Tom Fawcett. An introduction to roc analysis. *Pattern Recogn. Lett.*, 27:861–874, June 2006.
- [10] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27:1615–1630, October 2005.
- [11] Krystian Mikolajczyk and Cordelia Schmid. Indexing based on scale invariant interest points. In *Proceedings of the 8th International Conference on Computer Vision*, pages 525–531, 2001.
- [12] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision*, volume 2, pages 1508–1511, October 2005.
- [13] Edward Rosten, Reid Porter, and Tom Drummond. FASTER and better: A machine learning approach to corner detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 32:105–119, 2010.
- [14] Motilal Agrawal, Kurt Konolige, and Morten Rufus Blas. CenSurE: Center surround extremas for realtime feature detection and matching. In *European Conference on Computer Vision*, pages 102–115, 2008.

- 
- [15] Mosalam Ebrahimi and Walterio Mayol-Cuevas. SUSurE: Speeded up surround extrema feature detector and descriptor for realtime applications. In *“Workshop on Feature Detectors and Descriptors: The State Of The Art and Beyond” as part of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2009.
  - [16] Intel® SSE4 programming reference: <http://software.intel.com/file/18187>. page 156, 2007.
  - [17] Florica Mindru, Tinne Tuytelaars, Luc Van Gool, and Theo Moons. Moment invariants for recognition under changing viewpoint and illumination. *Comput. Vis. Image Underst.*, 94:3–27, April 2004.
  - [18] Anna Bosch, Andrew Zisserman, and Xavier Muñoz. Scene classification using a hybrid generative/discriminative approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30:712–727, April 2008.
  - [19] Joost van de Weijer, Theo Gevers, and Andrew D. Bagdanov. Boosting color saliency in image feature detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28:150–156, January 2006.
  - [20] Alaa E. Abdel-Hakim and Aly A. Farag. CSIFT: A SIFT descriptor with color invariant characteristics. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, pages 1978–1983, Washington, DC, USA, 2006. IEEE Computer Society.
  - [21] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conference*, volume 1, 2002.
  - [22] G. Bradski. The OpenCV Library. *Dr. Dobbs's Journal of Software Tools*, 2000.
  - [23] Mohamed Aly, Peter Welinder, Mario Munich, and Pietro Perona. Scaling Object Recognition: Benchmark of Current State of the Art Techniques. In *First IEEE Workshop on Emergent Issues in Large Amounts of Visual Data (WS-LAVD), IEEE International Conference on Computer Vision (ICCV)*, September 2009.
  - [24] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, pages 2161–2168, Washington, DC, USA, 2006. IEEE Computer Society.